Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER (9)
4. TITLE (and Subtitle) (6) SUPPLEMENTAL CONCEPTUAL DESIGN STUDY OF AN INTEGRATED VOICE/DATA SWITCHING AND MULTIPLEXING TECHNIQUE FOR AN ACCESS AREA EXCHANGE		5. TYPE OF REPORT & PERIOD COVERED Final Report, Aug[redacted] through Oct[redacted]76
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s)		8. CONTRACT OR GRANT NUMBER(s) (15) DCA100-75-C-0071 NEW
9. PERFORMING ORGANIZATION NAME AND ADDRESS GTE Sylvania Incorporated Electronic Systems Group, Eastern Division ✓ 77 A Street, Needham Heights, Mass. 02194		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS PE33126K, Task 13103G
11. CONTROLLING OFFICE NAME AND ADDRESS Director, Defense Communications Agency Courthouse Road Arlington, VA 20305		12. REPORT DATE (11) 11 Nov[redacted]76
		13. NUMBER OF PAGES 204
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) (12) 207p.		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Approved for Public Release, Distribution Unlimited.

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Integrated voice and data switching	Digital Access Exchange (DAX)
Unified transmission and multiplexing structure	Processor Architecture
Flexible time division multiplexing	Software Structure
Dynamic allocation of Channel Capacity	
Slotted Envelope Network (SENET)	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The overall Phase I SENET-DAX study was directed towards investigating the concept of an integrated voice/data switching system based on dynamic allocation of variable-sized increments of a flexible time division multiplexing transmission system. This permits a unified transmission and multiplexing structure that exhibits a high level of transmission efficiency not obtainable by separate voice and data systems, while supporting widely disparate types of communications.

Unclassified 406 451

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

The purpose of this supplemental report is to analyze further the integrated voice/packet/data transmission concept established in the Phase I effort, and to determine how it may be realized from generally available hardware, software, and firmware processing modules. Alternate conceptual system designs for an access area exchange model were investigated and an experimental approach to examine the integrated voice/data switching and multiplexing technique for access area application was formulated. In contrast to Phase I, the conceptual designs developed in this report emphasize processor architecture and software structure for an experimental model whose primary purpose is to demonstrate the workability of the basic SENET-DAX transmission concept in a realistic environment.

FINAL REPORT

SUPPLEMENTAL CONCEPTUAL DESIGN STUDY OF AN INTEGRATED VOICE/DATA SWITCHING AND MULTIPLEXING TECHNIQUE FOR AN ACCESS AREA EXCHANGE

11 November 1976

Submitted to
Director
Defense Communications Agency
Courthouse Road
Arlington, Virginia 20305

Contract No. DCA100-75-C-0071

Submitted by

GTE SYLVANIA
INCORPORATED
ELECTRONIC SYSTEMS GROUP–
EASTERN DIVISION
77 "A" STREET
NEEDHAM HEIGHTS, MASSACHUSETTS 02194

406 451

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Cont)

LIST OF TABLES

LIST OF TABLES (Cont)

SECTION 1

INTRODUCTION

1.1 PURPOSE AND SCOPE OF STUDY

The overall SENET-DAX study is directed towards investigating the concept of an integrated voice/data switching system based on dynamic allocation of variable-sized increments of a flexible time division multiplexing transmission system [3,4,27]. This permits a unified transmission and multiplexing structure that exhibits a high level of transmission efficiency not obtainable by separate voice and data systems, while supporting widely disparate types of communications. Since the primary cost of many military communication networks, particularly intercontinental and global communication networks, is in the transmission segment, integrating a variety of differing traffic types in this manner can lead in the future to communication systems that are economically practical.

During the Phase I portion of the study [26], the feasibility of the SENET-DAX concept in terms of structure and performance characteristics was analyzed and the concept was translated into representative hardware and software techniques that could be used for the conceptual design of an access area exchange. One system architecture, the distributed microprocessor/memory approach to the SENET-DAX concept, was chosen as the recommended implementation based upon the advantages accruing from its functional and physical partitioning and the corresponding independent operation of its constituent processing elements.

The purpose of the portion of the study described in this report has been to analyze further the integrated voice/packet/data transmission concept established in the Phase I effort, and to determine how it may be realized from generally available hardware, software, and firmware processing modules. Alternate conceptual system designs for an access area exchange model were investigated and an experimental approach to examine the integrated voice/data switching and multiplexing technique for access area application was formulated. In contrast to Phase I, the conceptual designs developed in this report emphasize processor architecture and software structure for an experimental model whose primary purpose is to demonstrate the workability of the basic SENET-DAX transmission concept in a realistic environment. In addition, the designs were to maximize the use of hardware, firmware, and software processing modules presently available or expected to be widely available in the future without special design. The basis for the architectural designs to be considered were those established in accordance with the Phase I study effort and, as available at the time of contract initiation, other conceptual designs accomplished through other independent and related DCA studies.

1.2 TECHNICAL APPROACH

The following technical approach was used during the course of the study. As a consequence of the design experience acquired during the Phase I study, it was possible to define fundamental SENET-DAX concepts and to identify features necessary in order to demonstrate workability of the concept in a realistic environment (e.g, in the presence of noise). Next, several alternative system designs were developed, each of which was capable of demonstrating the workability of the basic concept. Among these were: the class of systems using a distributed processor architecture, the configuration recommended in Phase I; a central processor architecture; and a multiprocessor/multimemory interconnect architecture as described in [25]. Section 2 describes the various approaches to the processing architecture and associated software structure which were considered here for the access area exchange model. In Section 3 the most applicable and cost-effective of these designs, the distributed model, was recommended for the access area exchange. The basis for this choice was derived from several factors. One was a timing analysis that was made for certain basic time-critical software routines in the model, e.g., real-time clock interrupt and packet analysis processing. These software functions are time-critical in the sense that most are repetitive and require real-time or close to real-time processing capability. Our conclusion for the recommended architecture was further supported by data obtained from the equipment/software survey, evaluation of link memory access capability and estimates of system cost. The distributed model was specified first in functional block form, where each block is associated with the functions it performs. For example, link hardware performs the functions of start-of-frame detection, bit stuffing, redundancy error checking, and packet flag detection. In the next step, a more detailed design was specified. It was from this level of detail that we based our estimated system costs.

In Section 4 we report the results of an equipment/software survey, which was undertaken to identify the available processing modules to be used as building blocks in the recommended conceptual design. Vendor literature was solicited and several vendors were brought in for meetings. Survey areas that related to equipment included size, availability and cost, along with diverse operational capabilities and alterability within an experimental environment. The software investigation dealt with the evaluation of instruction sets, availability of program development and systems support from the manufacturer, and program development costs.

An approach which can be used to examine experimentally the integrated voice/data switching and multiplexing technique is described in Section 5. Several alternative access area exchange configurations were evaluated toward an experimental network for system test and evaluation. The basis of choice was the capability to provide the features necessary to demonstrate the workability of the SENET-DAX concept in a realistic environment, yet be cost-effective as well. The recommended network for system testing is described more fully in paragraph 1.3.

A performance evaluation of the merits of the recommended conceptual design, the distributed processing model, was made as a function of switching capabilities, and the results were reported in Section 6. Among those performance criteria evaluated were transmission efficiency and throughput as a function of packet size, bit error rate, and error control protocol; blocking and delay of voice and data; system traffic and sizing characteristics; trunk handling capability; and cross office and cross network delays. Results are shown in a parametric fashion where possible.

Cost estimates for the recommended conceptual design model and the experimental net orks compored of these models are provided in Section 7. The impiementation costs for the distributed architecture model were derived from the detailed hardware diagram developed in paragraph 3.3. Costs are itemized on the basis of hardware estimates only; engineering and software development costs are not included (except for certain development aids as they relate to the hardware). Cost charts are provided and offer a relative comparison of network costs as a function of network complexity.

Section 8 concludes the report with a recommendation for further investigation into certain areas which have great potential application toward an experimental SENET-DAX access area exchange multi-node network. These include communication over satellite links and interface with the ARPA network.

1.3 EXPERIMENTAL/DEMONSTRATION NETWORK

The ultimate objective of this study effort is to be able to design, build and test experimental models in equipment and software that will demonstrate the practicality of the SENET-DAX transmission concept in a realistic network environment. To this effect, integrated voice/data switching centers are configured in a complex network organization for system test and evaluation. In the system organization, only the processing power, memory capacity and terminal service capability necessary to explore the concept for a working model based on the SENET-DAX concept are provided.

The basic SENET-DAX concepts to be demonstrated include:

a. Integration of voice (Class I) and data (Class II) and their interaction in a single switching system

b. Dynamic allocation of channel capacity (voice and data) in variable sized increments of the Class I and II regions to provide efficient and non-interfering communication through the switched trunk network. This includes dropping of channels for new calls, and recompacting the frame after detection of call release

c. Servicing of voice subscribers utilizing different voice rates (various channel widths) and a variety of current inventory and development equipment

d. Servicing of data terminals with different control philosophies, e.g., interactive graphic and query/response terminals using packet switching techniques, and narrative and bulk data terminals using store and forward techniques

e. Movable boundary between classes of data where packet-switched data traffic is allowed to occupy transmission capacity not used by circuit-switched voice traffic, and vice versa.

Table 1-1 identifies the services necessary to demonstrate these fundamental concepts in a realistic environment.

A number of network configurations were considered for the experimental model that could demonstrate the fundamental SENET-DAX concepts and provide the services listed in Table 1-1. Figure 1-1 illustrates an experimental four-node network with the connecting links operating full-duplex at a rate of 230.4 kBPS. Each node consists of a SENET-DAX switch with tandem capability and a limited number of directly connected digital subscribers.

TABLE 1-1. SENET-DAX SERVICES TO DEMONSTRATE WORKABILITY OF THE BASIC CONCEPT

Multiple voice rates

Dynamic allocation of Class I region including drop, insert, and recompacting data

Integrated voice/data with movable boundary

Call setup with forward reservation and backward allocation

Call release

Common channel signaling

CCIS artificial precedence

Precedence and preemption of voice and data

- Multiprecedence in a packet network
- Preemption-Class I data by Class I
- Delay-Class II data by Class II
- Precedence reconciliation between data classes

ADCCP procedures for data & CCIS

- Packetizing/depacketizing data and CCIS messages
- Data/CCIS error control

Protocol processing

- Link protocol
- CCIS protocol
- Originating/terminating DAX protocol

Dynamic alternate routing

Voice and data terminal interfaces

Network synchronization

Frame synchronization/resynchronization

Linked list processing

Satellite delay modelling

Data terminal throttling

Voice service includes 10 to 12 subscribers, primarily 16 kBPS and 32 kBPS CVSD terminals, with possibilities for 2400 BPS vocoders, and 50 kBPS KY-3 voice terminals. Only compatible (technique and/or transmission rate) voice terminals will be allowed to converse; no voice rate conversion between non-compatible terminals will be incorporated. Data terminals will include teletypes at different baud rates (110, 150, and 300 baud), facsimile terminals, and possibly computer terminals.

The trunk rate of 230.4 kBPS is a commercial common carrier standard and is available for rental from the telephone company. This choice of a standard transmission rate seemed a practical solution to the problem of using a lower carrier rate than was used in Phase I, but having the capability of handling sufficient traffic to demonstrate the SENET-DAX concept. Figure 1-1 shows a maximum of three 230.4 kBPS trunks at a given node. However, to allow for expandability, each node in the network was designed for a maximum capability of handling five trunks.

This network will support the demonstration of all the basic services listed in Table 1-1, including recovery from blocking and full alternate routing capability. The model will accommodate satellite radio links between switching nodes. For laboratory purposes satellite delays may be simulated by delay lines.

In order to allow network links to be occupied by a wide distribution of data for experimental purposes, a table driven traffic generator will be designed as part of the access memory. Each switch will thus have the capability to generate traffic to occupy trunks without requiring the use of terminals. This feature provides added flexibility in generating voice calls and data messages of varying bit rates without having to provide more data terminals and hence increase system complexity. Class I calls established via the traffic generator would use CCIS procedure to set up and break down the calls between switches, but would not employ subscriber signaling (ringback, ring forward, dial tone, etc.), since subscribers would not exist for these calls.

SECTION 2

SYSTEM DESIGN ALTERNATIVES

2.1 GENERAL

Systems utilizing multiple processors are today beginning to see increased application in a number of environments that only a few short years ago would have been implemented by large monolithic systems. The traditional problems of multiprocessor systems have been well documented and include considerations of cost, computational capability, large complex system software, memory access conflicts and methodologies for parallel task execution.

In carrying forward the exploration of the Phase I results toward the suitability for modeling, and in view of the above complexities of multiprocessor architecture, this study focused early on an objective review of alternative designs so that maximum advantage could be taken of reduced model requirements. Three candidate system designs were developed and examined with respect to their capability to demonstrate the workability of the basic concept and for factors relating to their reliability and software complexity. Each design necessitated sufficient investigation to determine its suitability or dismissal when weighed against established design goals and desired capabilities of the model. Other multiprocessor configurations referenced in the bibliography, were reviewed for applicable technology and methodology adaptable to this application. [7,18,22].

2.2 CONCEPTUAL DESIGN MODELS CONSIDERED IN THE STUDY

The system designs considered in this study include:

a. A multiprocessor/multimemory interconnect structure based upon the Carnegie-Mellon Cmmp report

b. A central processor approach

c. The distributed micro-computer approach as recommended in Phase I.

The models are specified in functional block form in the following sections.

2.3 MULTIPROCESSOR/MULTIMEMORY INTERCONNECT STRUCTURE

Figure 2-1 displays an architecture based upon the multiprocessor configuration presented in the Carnegie-Mellon report. With reference to this figure, certain advantages may be identified. These advantages include the ability to gracefully degrade performance as the result of the failure of a task processor. Under this condition, all trunks can continue to be serviced at a reduced rate. It should also be noted that this system exhibits modular growth in terms of trunk and access servicing.

However the architecture cannot be considered a viable candidate system from the point of view of required reliability. There are a number of single thread elements whose redundancy would have an impact on system cost and complexity.

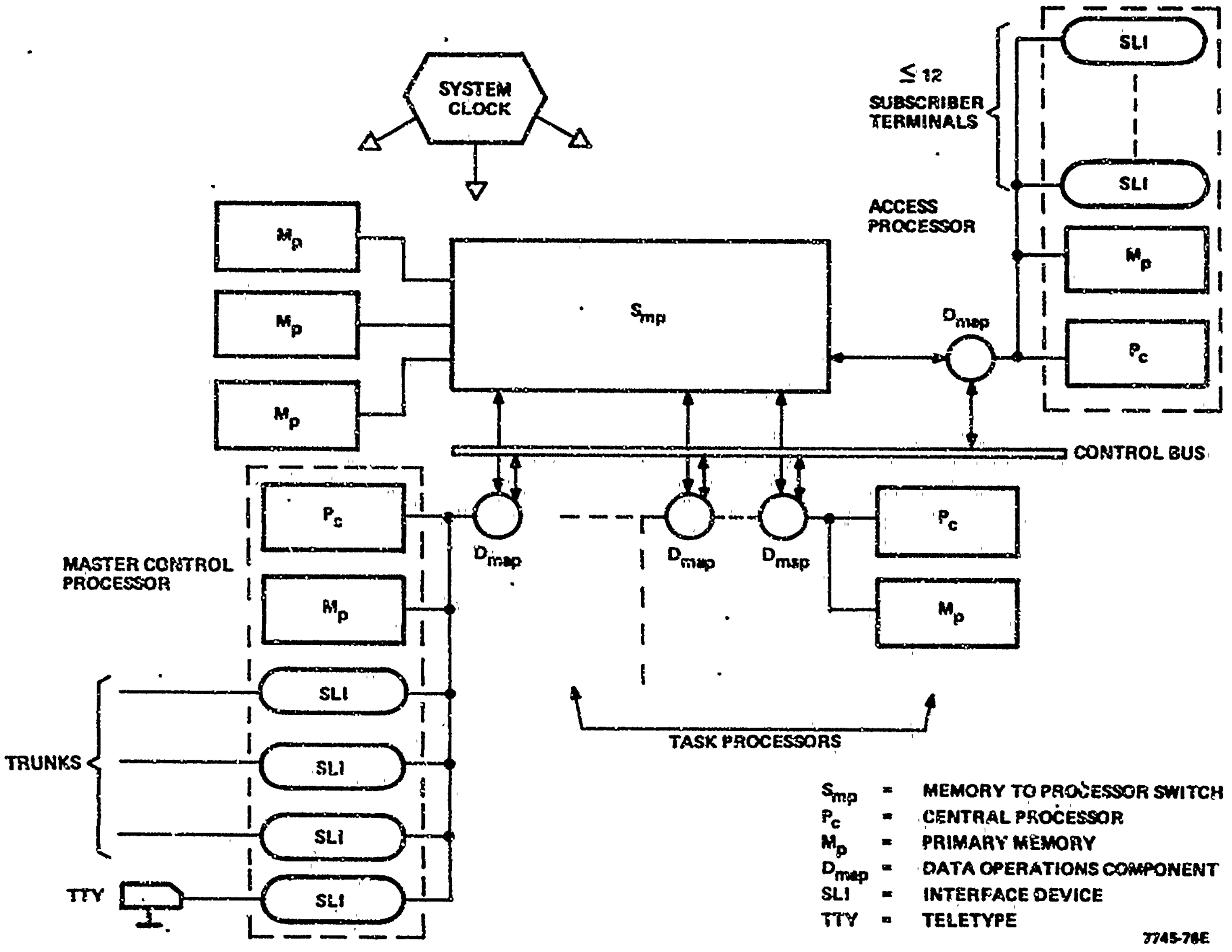


Figure 2-1. Multiprocessor/Multimemory Interconnect Structure

The master control processor responsible for task decomposition and task assignment to other processors is itself a single thread element along with the primary memory switching/addressing element (Smp) and control bus. Methodologies for task allocation for pipeline processing are not well established and the number of processors would require a complex system software control structure that would have a negative effect on system performance. Since the number of tasks is independent of the site size, this configuration exhibits high cost for smaller sites and anticipated poor performance at larger sites due to increased memory access conflicts. These overriding disadvantages precluded further detailed consideration of this architecture in view of more promising alternative configurations.

2.4 CENTRAL PROCESSOR APPROACH

The impetus for consideration of this approach stemmed from the desire to take advantage of any relaxation of system requirements achieved in defining the design goals and capability of the experimental model. Two areas where such relaxation is achieved are in the proposed transmission speed and in the number of representative subscribers necessary to validate the design experimentally. In the latter case, provision for the following set of typical terminal rates insures adequate representation consistent with reasonable system test flexibliity and interface costs.

Typical Terminal Rates

110 Baud	Assumed loading of up to 12 subscribers per node.
2.4 kBPS	
9.6 kBPS	
16.0 kBPS	
32.0 kBPS	
50.0 kBPS	

Common carrier equipment sufficient to handle the above aggregate subscriber rate is commercially available at a line rate of 230.4 kBPS and is therefore proposed as the transmission speed for our model. This represents the second area where relaxed performance requirements suggest that a centralized approach may be utilized in order to demonstrate the concepts of the SENET-DAX system.

Further examination of the interrupt service rate required by a bi-directional 50 kBPS subscriber reveals that an 80 μsec processing interval is all that is available to service/process incoming and outgoing data bytes. A single processor cannot provide software servicing at these rates even if that were all it was required to perform. For this reason, the configuration shown in Figure 2-2 is separated into two processors in order to off-load real-time subscriber access onto the digital access controller. In this arrangement, network processing such as list management, routing and packet analysis/generation must be performed by the trunk processor.

There are two outstanding considerations inherent in this approach. First, the trunk processor must be capable of concurrently processing the input and output network functions, and second, the data memory must be operated as a shared resource to avoid the internal transfer of memory data. Sharing this resource imposes the necessity of a high speed control information transfer between processors to coordinate the placement and extraction of data which adds to the processing time constraints of both processors. Reliability, redundancy and switchover considerations for single thread elements create additional system cost and complexity as they did in the multiprocessor/multimemory interconnect structure discussed in paragraph 2.3.

It is interesting to note that relief for the concurrent input/output processing requirement of the Trunk Processor, shown in Figure 2-2, may be obtained by further subdivision of this processor into multiple processors. This strongly suggests that this case is a simplex form of the distributed architecture when the I/O speeds are such as to allow the contraction of all processing into a singular element.

Multiple processing elements again raise the question of sharing a common memory resource and whether a multiprogramming approach to enhance CPU utilization, degraded by memory access conflicts, is justified.

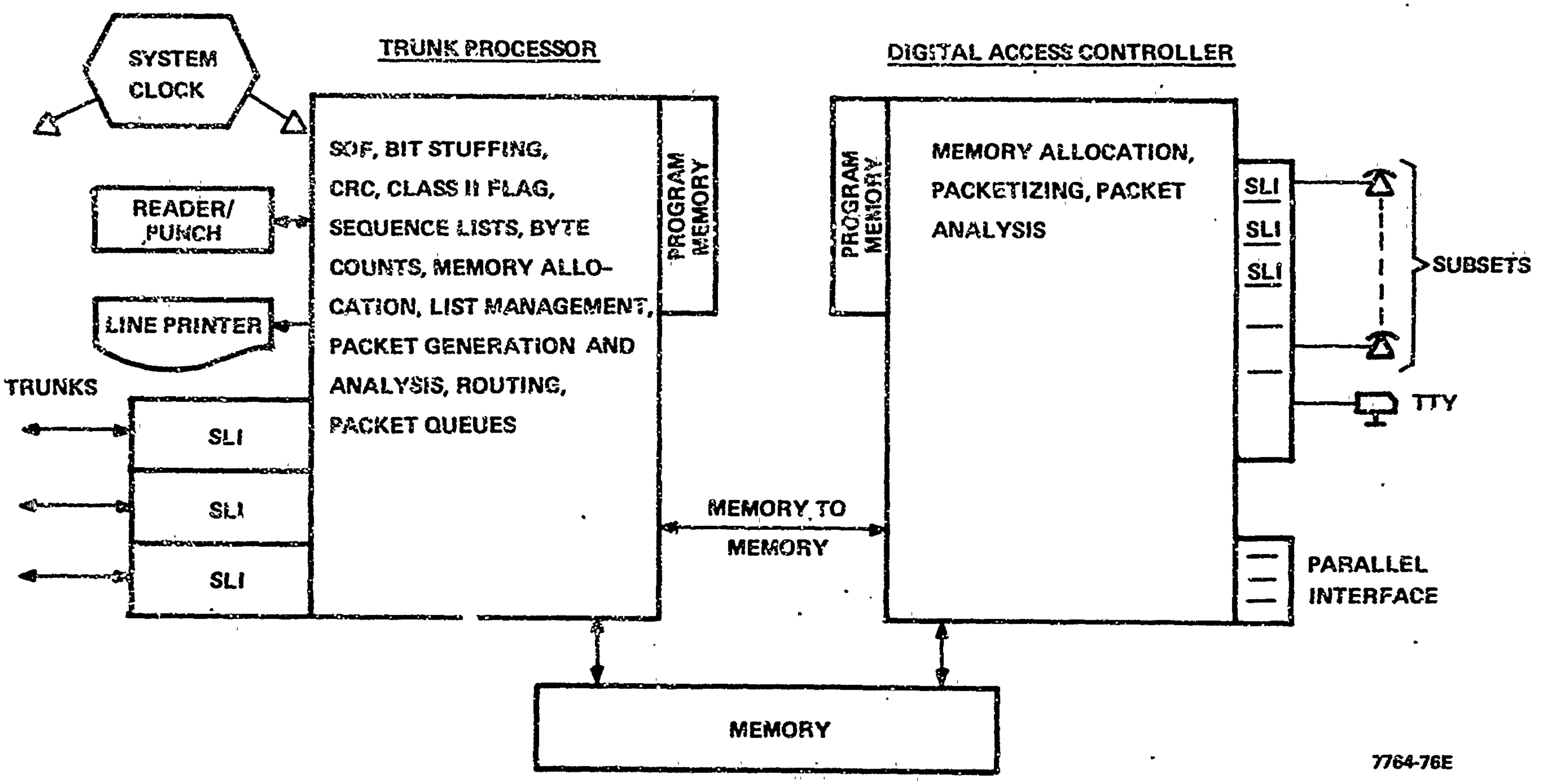

SYSTEM CLOCK
TRUNK PROCESSOR
DIGITAL ACCESS CONTROLLER
READER/ PUNCH
LINE PRINTER
TRUNKS
SLI
SLI
SLI
SOF, BIT STUFFING, CRC, CLASS II FLAG, SEQUENCE LISTS, BYTE COUNTS, MEMORY ALLOCATION, LIST MANAGEMENT, PACKET GENERATION AND ANALYSIS, ROUTING, PACKET QUEUES
PROGRAM MEMORY
PROGRAM MEMORY
MEMORY ALLOCATION, PACKETIZING, PACKET ANALYSIS
SLI
SLI
SLI
SUBSETS
TTY
PARALLEL INTERFACE
MEMORY TO MEMORY
MEMORY
7764-76E

Figure 2-2. Divided Central Processor Configuration

2.5 SHARED MEMORY CONSIDERATIONS

A review of computer memory systems will show that the requirement for a shared memory resource grew out of a desire to increase CPU utilization by allowing peripheral I/C direct access to the memory without processor involvement. Earlier computer systems labored under limited memory technologies which mainly divided between electronically addressable (core, semi-conductor) and electromechanical (tapes, drums) devices which created a disparity in accessing time in any given memory hierarchy.

Under this environment, multiprogramming concepts evolved in order to continue task processing during resource access intervals, assuming that the access time was sufficiently lengthy to justify the overhead involved in task switching. The common conclusion of these observations is that multiprogramming and shared memory techniques appear justified only when the system must deal with "access gaps" due to different memory technologies and I/O equipment (characteristics not present on our experimental model).

Today, memory technology is making rapid progress in reducing the differences between hierarchal information access times with such advances as charge-coupled devices (CCD's), bubble memories with a potential of 10 million bits per square inch, electron beam addressed memories (EBAM) and domain type propagation (DOT), [1,15,19,20]. The proposed architecture for the SENET-DAX model places this system in a position to readily adopt these new technologies as they become available for application in the next decade, since multiprogramming complexities have not been introduced into the conceptual SENET-DAX design thereby enabling future system enhancement with minimum complication.

Currently, 100 nanosec access time memories represent state-of-the-art. Memory access requirements must therefore receive careful consideration in the selection of a memory hierarchy, since in the application being considered all devices peripheral to memory have equal priority and service time requirements. Under these conditions, a shared memory approach reduces to a FIFO (first-in, first-out) service algorithm with contention resulting in system delays. The inpact on memory access requirements with zero conflicts for a distributed architecture may be compared with those of a shared memory approach and this comparison is shown in Figure 2-3.

With reference to this figure, it should be noted that the calculations were done for the experimental model transmission speed stated earlier (230.4 kBPS) and assume that the memory accesses are not only a function of the number of processors (and therefore the number of links) but are also a function of the number of accesses required for processing the stored data. The number of processing accesses (Ap) is related to the degree of software processing complexity, with Ap=0 parametrically establishing the lower bound when no useful work (processing) is performed other than storage and retrieval. It can be seen that for a model structure consisting of 3 links the operating range of the memory will be between 360 nanosecs

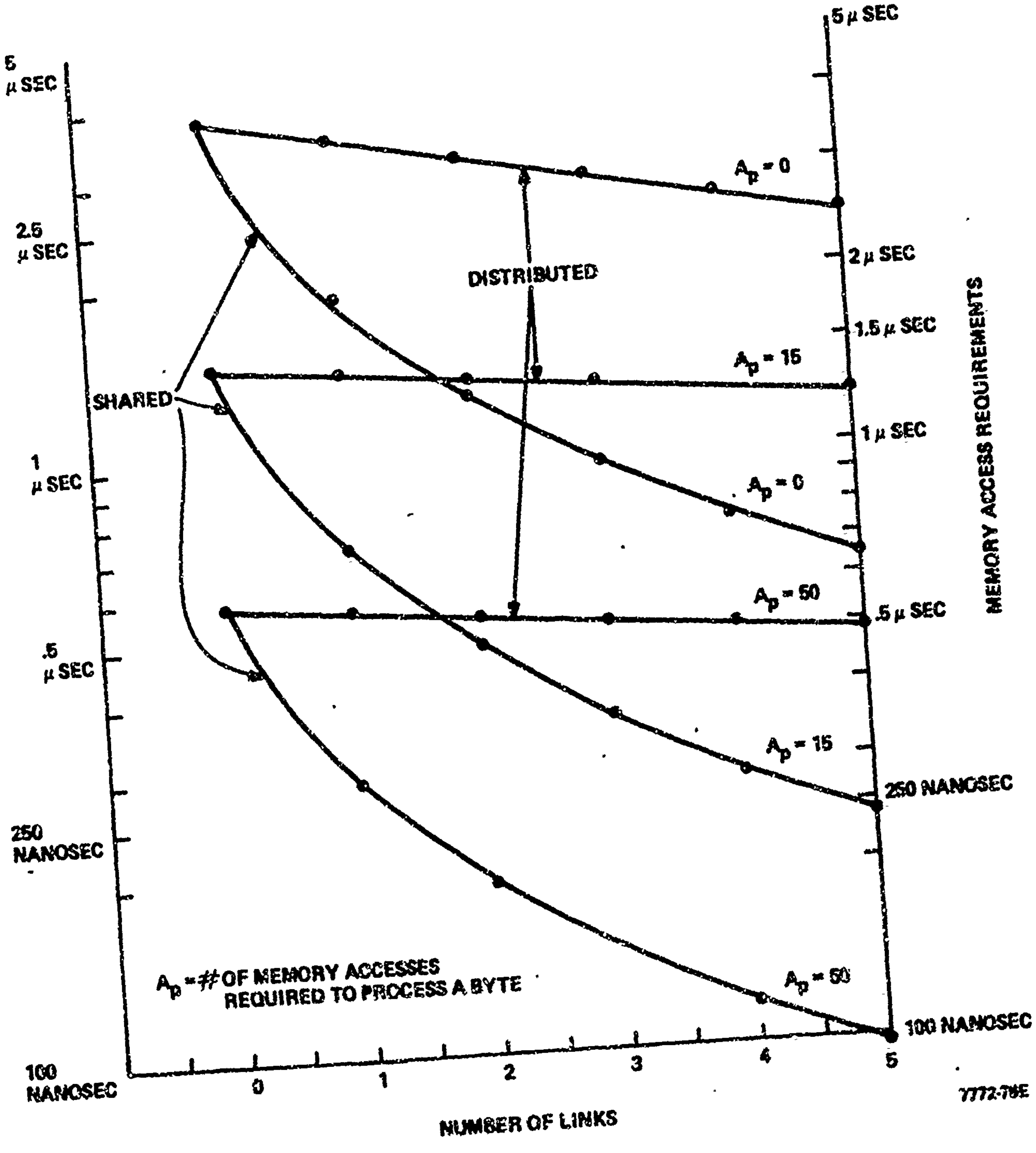


Figure 2-3. Comparison of Memory Access Requirements for the Distributed vs. Shared Architecture

with Ap=15 for the shared system and 2.9 μsecs for Ap=0 for the distributed approach. Both of these extremes lie well within current technology for implementation of a scaled prototype system.

However, in order to derive meaningful experimental data, our model architecture must closely approximate that of the final system wherein the maximum number of links may reach 15 with all links operating at T1 carrier rates (1.544 Mbps). Due to the different transmission speed (T1 carrier) these points could not easily be shown on the same plot. Calculation of the memory access requirements for this case was performed separately and show that for a 15-link system the access time requirements, for the distributed approach, fall in the range of 133 nanosecs to 216 nanosecs, depending upon the complexity of processing. Parameter Ap was allowed to vary in this case from zero (no processing) to a maximum of 15 (accesses per byte per processor).

The results of a similar calculation for the shared memory reveals that even for an Ap=0, the memory access requirement would be 36 nanosecs while an Ap=15 produces a requirement of 13.5 nanosecs, indicating that this approach would clearly exceed current state-of-the-art memory technology.

It can therefore be concluded that in a system with equal priority and service time requirements, neither shared memory systems nor systems involving a high level of multiprogramming provide the degree of performance exhibited by distributed memory systems. In a system of equal priorities and an era of rapidly advancing memory technology in which there is no question that the future for memories is one of higher densities, faster speeds and lower cost per bit, we are firmly convinced that distributed memory systems will permit higher performance with complex processing for a given state-of-the-art and more easily permit the adaptation of new memory systems as these become available.

2.6 DISTRIBUTED SYSTEM ARCHITECTURE

A system concept incorporating distributed processors coupled with distributed data memories is characteristically one in which data throughput, application versatility and system reliability are optimized through the modular partitioning of system functions. Whereas a multiple processor/memory interconnect system through parallel processing increases the data throughput rate while providing graceful single failure degradation characteristics and distributed diagnostic capabilities, system overhead can be large due to the supervision required for high speed interprocessor communications. This overhead can be of sufficient complexity as to compromise much of the throughput advantage.

In an attempt to maximize all desirable characteristics of a distributed system, the distributed architecture illustrated in Figure 2-4 has been developed. This system is characterized by a dual bus structure with a continuous cycle bus scheduling clock. The total interprocessor communications overhead function is performed simply by a clock driven multiplexer which multiplexes all data and control bus transfers. For processor-to-data memory access across the bus, delay

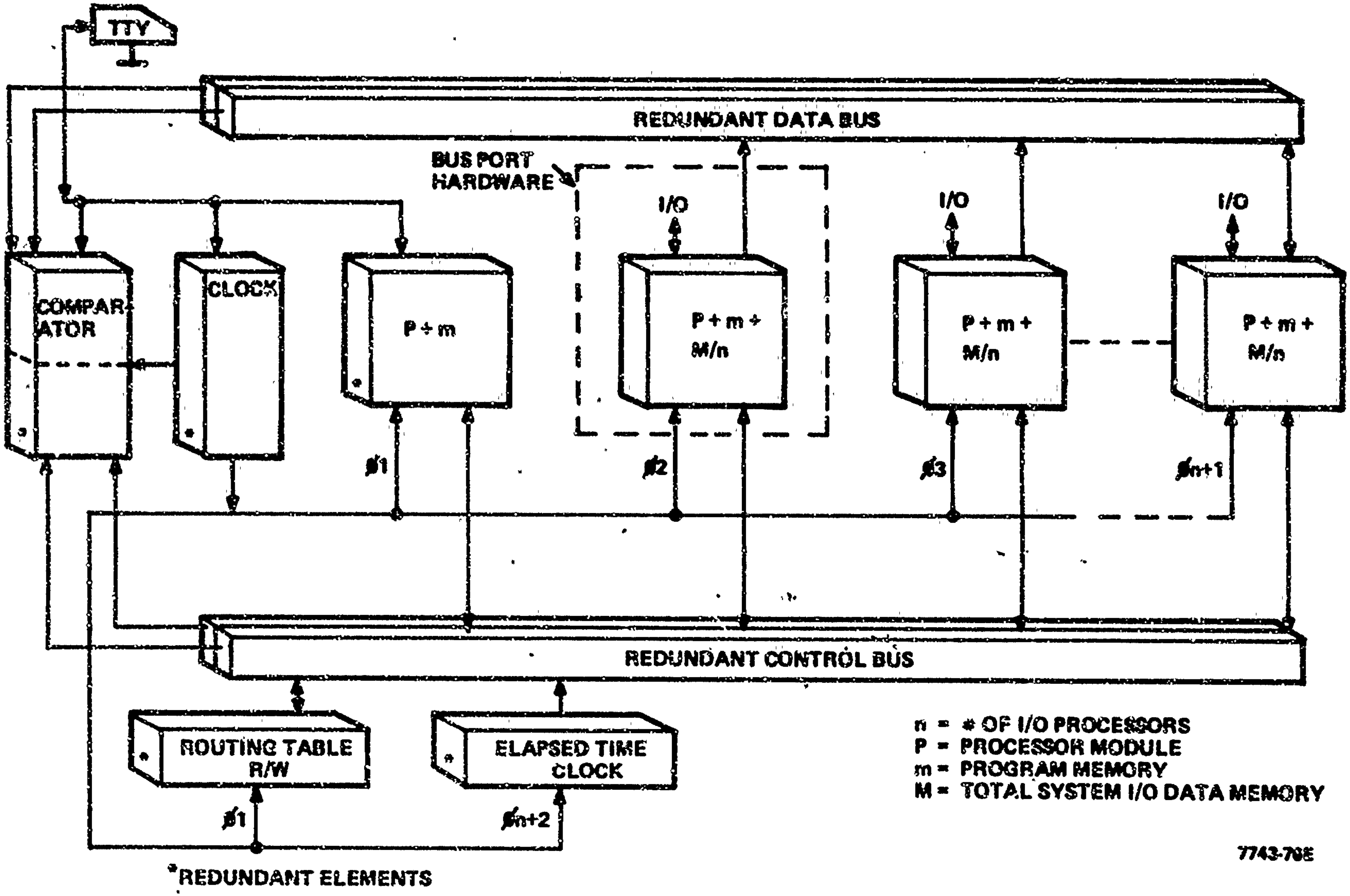


Figure 2-4. Distributed System Architecture

is minimized by assuring port-to-data bus access at least once per byte time based on the highest rate I/O transfer. Real-time bus scheduling for port hardware eliminates priority rating and interrupt contention delays. At each bus port, a first-in-first-out buffer collects chronological control bus transfers, which were addressed to that port, for time ordered precedence processing of required operations.

The bus port hardware indicated in Figure 2-4, which is incrementally expandable on the buses, contains one or more processors, a portion of system data memory and customized I/O hardware (Figure 2-5). The modular independence of both the port processor and the associated data memory provides high system reliability. System reliability is further enhanced by protecting all port data memories from erroneous modification. This is accomplished by allowing only a port input processor to alter the contents of its associated data memory.

Generally, this distributed system architecture also provides a high degree of application versatility. In view of the modular programmed control per port, a wide variety of I/O circuit or lines may be serviced simultaneously by one system. Depending on the type of I/O servicing required at a bus port, the programmed element utilizes a communications interface module for direct I/O control or to perform a more supervisory function over remote operation.

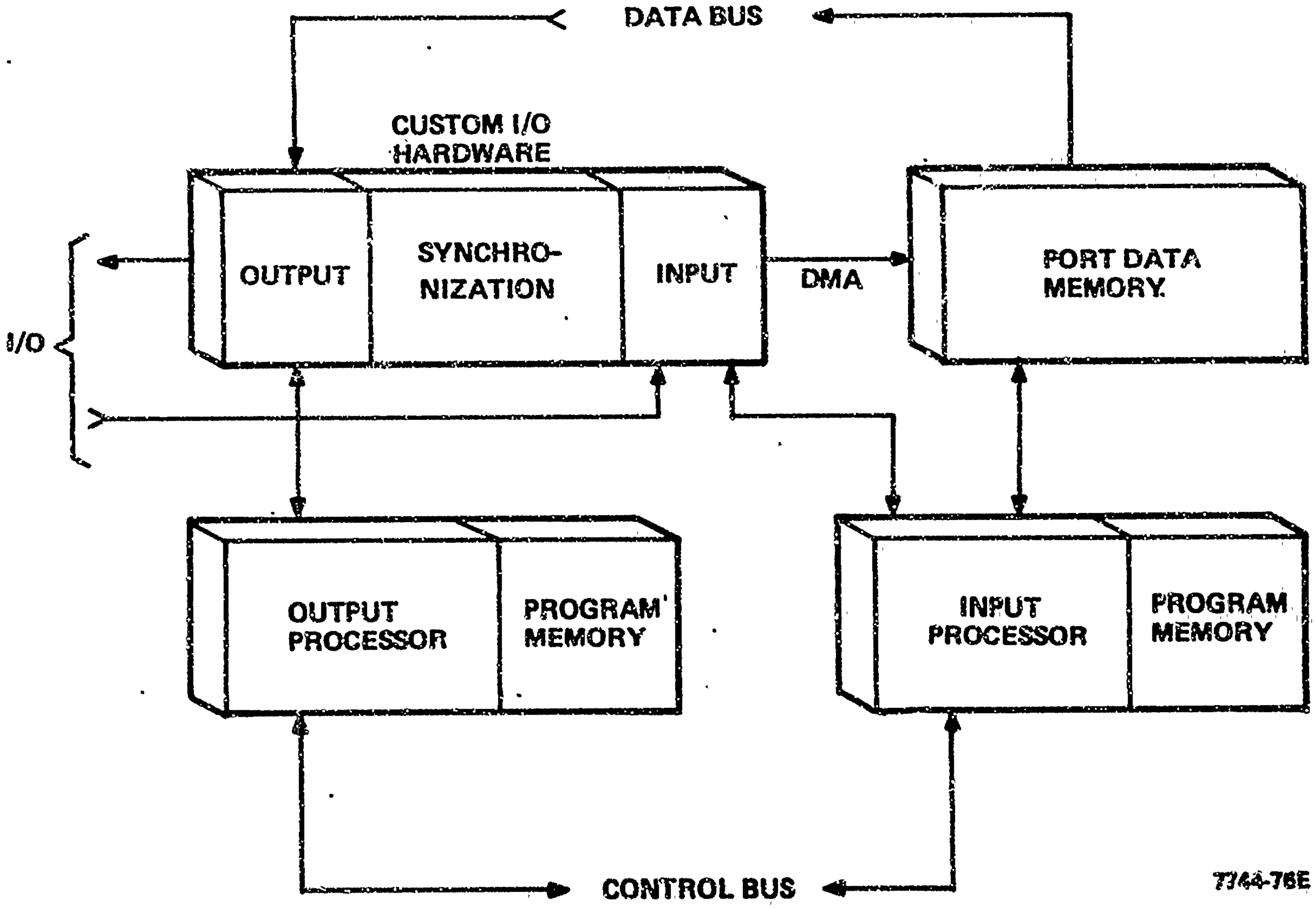


Figure 2-5. Bus Port Hardware

The distributed system architecture presented herein provides a system method for satisfying the extensive requirements of high data throughput speed, operational and application versatility, and reliability. These desirable system characteristics are facilitated by the modular partitioning of system functions, coordinated via a uniquely controlled bus structure.

2.7 OTHER DESIGN CONFIGURATIONS

Although time did not permit an in-depth study of all systems in the periphery of multiprocessor configurations, two additional systems were reviewed for their architectural applicability to the SENET-DAX concept. [7,18].

The associative processing technique put forth by Honeywell Systems/Research Center provided for the compression and multiplexing of digitized voice and data by individually assigned line processors. From an architectural standpoint, a common bus structure was employed between the master control processor and all line processors and as such is not directly applicable for the reasons stated earlier in this section on centralized processing.

The Pluribus configuration as described by Bolt, Beranak and Newman utilizes a pool of processors that are distributed along with their assigned memory elements, to a multiplicity of buses. Each basic building block is therefore composed of two processing elements, two memories and a bus with a standardized interface. Systems are configured by full interconnectivity of the individual buses with all processors equally available for task processing as tasks become available in a central hardware-controlled task queue. The modularity of this system would appear to make it extremely suitable for a wide variety of applications with apparent high reliability. It is our current observation that the present structure is based on a distribution of tasks akin to a multiprogramming structure which we do not believe to be well suited for the SENET-DAX system. It is also noted that memory access conflicts are resolved by having each bus contain its own conflict-resolution hardware. While a formidable amount of raw processing power appears to have been provided in a modular structure, inter-processor communication/handshaking appears to introduce internal delays along with the possibility for "glare" situations to develop. However, the complex combinatorial processing of such a system would require considerable further study prior to forming a definitive evaluation.

SECTION 3

RECOMMENDED DESIGN CONFIGURATION

3.1 GENERAL

The system design configuration recommended for modeling a SENET-DAX network must economically and operationally satisfy many diverse network and nodal criteria while demonstrating the SENET-DAX concept. These criteria include worse case operational demands, the need for minimal system overhead in limited service environments, modular expandability, high system reliability, automatic self-diagnostics, and the ability to statistically analyze and quantify operational procedures and events.

In the experimental laboratory environment, each network node will probably be required to service up to three high speed full-duplex trunk links, each at 230.4 Kb/s or less, with an upper limit of five links. At the access level, twelve terminations would be an assumed upper bound. Most of these access terminations would be synchronous, with rates up to 50 Kb/s. The remaining would be asynchronous with assumed maximum operating speeds of 9.6 Kb/s.

The worse-case operational demands based on the input/output rates above disqualify a central processor approach, (as will be discussed in paragraph 3.3). The processing time required to simply service the operational I/O would leave no time for diagnostics or statistics. In addition, the central processor approach for the application provides reduced system reliability, contains a maximum of system overhead, and does not allow for total modular expandability.

The Carnegie-Mellon multiprocessor system approach satisfies more system criteria than the central processor approach. This multiprocessor/shared-data-memory system, while allowing a greater degree of modular expandability, higher reliability due to the multiple processors, and distributed diagnostic and statistical capability, requires a considerable system overhead function. This overhead, which is necessary for the task scheduling of the system processors, significantly compromises system data throughput and limits real-time reaction speeds. System reliability and data throughput both suffer somewhat when a central shared memory is used.

Since the distributed system architecture utilizes a multiprocessor structure with distributed data memory and very low system overhead, this architecture is an interesting option. When it is also recognized that this architecture provides total modular expandability, high system reliability and a minimal system overhead which provides rapid no-contention interprocessor communication and data bus transfers, the distributed system architecture stands as the most promising alternative. Distributed diagnostics and statistical compilations are also realizable features in this system. Therefore, the distributed system architecture is recommended for the SENET-DAX model.

3.2 FUNCTIONAL DESCRIPTION OF THE DISTRIBUTED MODEL ARCHITECTURE

A functional block diagram of the SENET-DAX model distributed system architecture is presented in Figure 3-1. A two level fully bi-directional bus is the center of all nodal activity. Three hardware/software groups, which attach to this bus structure, are the nodal equipment group, the trunk link group, and the local access group.

The nodal group contains the system clocks, routing table, and nodal processor. All bus activity and system I/O are coordinated and synchronized by the system clocks. The nodal processor performs minimal supervisory control over nodal activities by, for example, updating the routing table and interfacing with a system attendant.

The bus, which has a total loading capacity of six bus ports, can service one local access group and a maximum of five trunk link hardware/software groups. A trunk link group services one full-duplex trunk link at a maximum rate of 230.4 Kb/s. All link protocol including synchronization is performed by the link processors. The link data memory, a portion of total nodal data memory, is supervised by the link input processor.

One bus port is used by a local access group. A maximum of eleven Class I and at least one Class II subscribers interface to the nodal bus structure via their respective processors. A customized DMA controller is supervised by both processors to facilitate the retrieval of data across the bus to be transmitted to local subscribers.

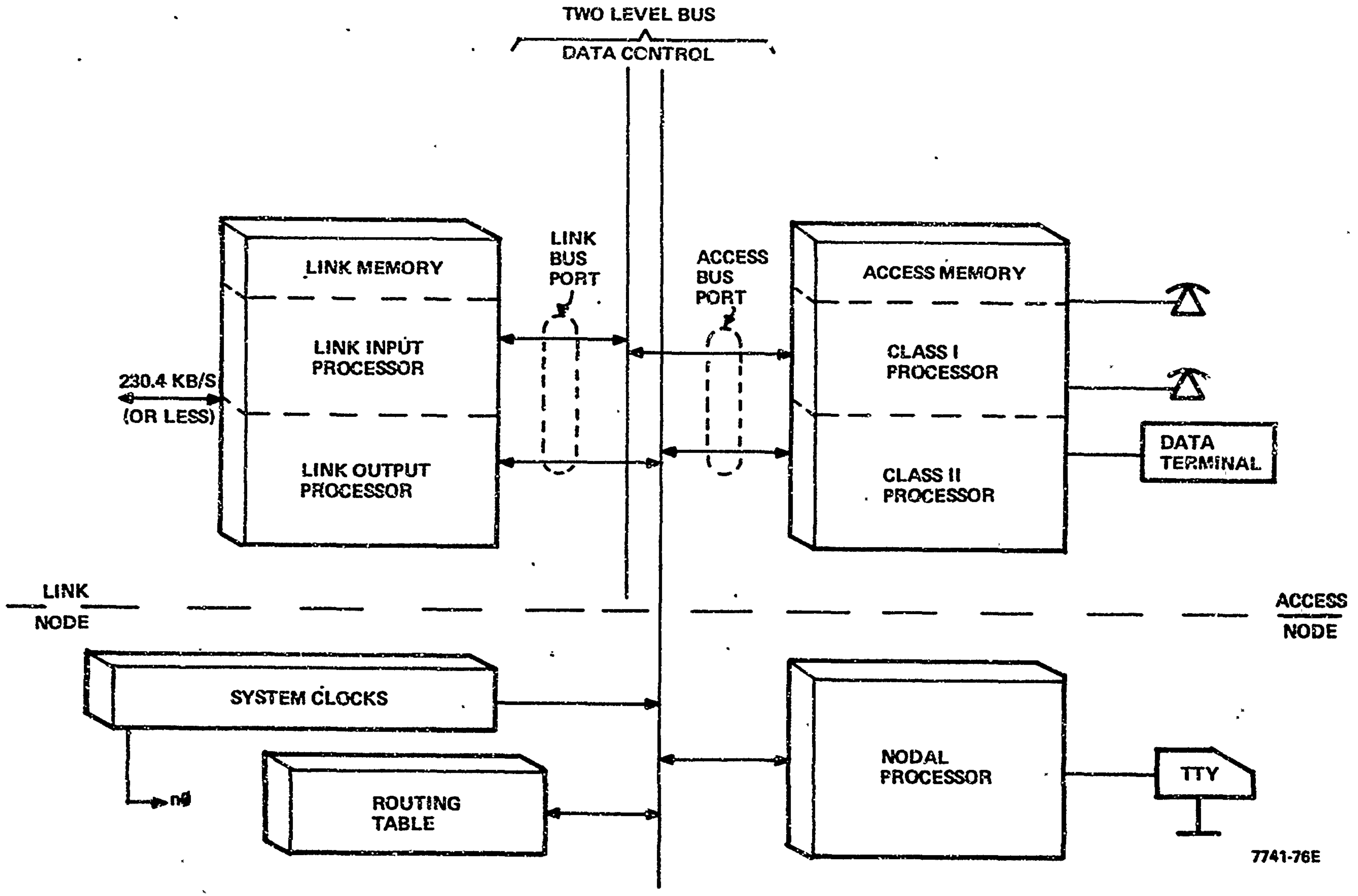


Figure 3-1. SENET-DAX Functional Block Diagram

3.2.1 Equipment Processing Elements

The recommended SENET-DAX model functional architecture is shown in Figure 3-2. The nodal processing element performs a remote supervisory function over all system activities by directing interprocessor communications via the routing table. The table contents are periodically updated by the nodal processor based on diagnostic and statistical reports received from all other system processing elements. All other system processing elements perform the primary task of interfacing a data input and/or output circuit to the central bus structure. A nodal clock multiplexer organizes all bus activities.

All processors have an attached program memory. One clock multiplexing phase is assigned to each processor during which it has complete command of the control bus. Each processor is also assigned one or more control bus addresses. For transfer of control data over the bus, the processor transmits a destination address with data onto the control bus. When an address is recognized by the appropriate first-in-first-out (FIFO) processor buffer (see Figure 3-2), specific control bus information is strobed into that buffer. Control FIFO's receive all control bus transfers related to nodal administration. Packet FIFO's receive transfers concerning the traffic at that port. These buffers facilitate the chronological task ordering of the interprocessor communications addressed to a processing element.

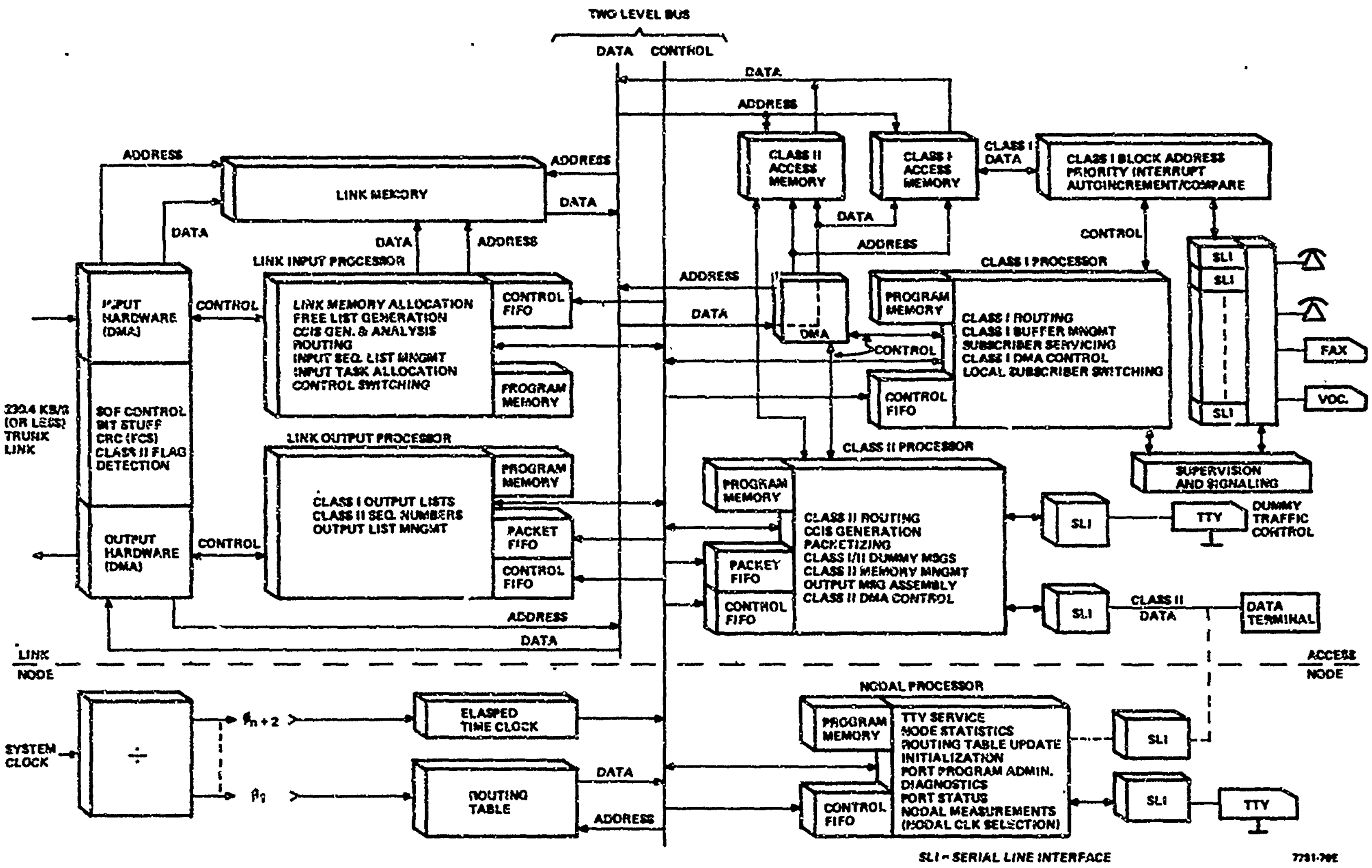


Figure 3-2. SENET-DAX Functional Architecture

The nodal processor interfaces an operator console to the control bus. Through this console the operator may exercise complete administrative control of any system program memory or collect traffic statistics on any system operation. If desired for network demonstration, the nodal processor may be connected at a Class II access-level data terminal connection to operate as a remote query-response computational facility.

At a link bus port, two processing elements are required to effectively service a trunk link. The link input processor manages the link data memory. After supervising an input hardware DMA byte transfer into link memory, the input processor analyzes the stored contents of Class II packets. Following an analysis, the input processor forms an acknowledge packet in its link data memory for transmission by its associated output processor.

The link output processor, after receiving traffic control information through its packet FIFO, incorporates the prepared packet into the transmitted serial data stream by instructing the output hardware to perform a DMA operation over the data bus for the duration of that packet. The packet is preceeded by a speciai flag and a sequence number, then appended by an error-checking cyclic redundancy code (CRCC) and another flag. All link synchronizing procedures including flags, CRCC and bit stuffing are performed by hardware associated with the link input/output hardware.

The access bus port also requires two processing elements. A Class II processor supervises all asynchronous subscribers and manages the Class II access memory. This includes the supervising of DMA byte transfers via the data bus into the Class II access memory from link memories. In addition, the Class II processor interfaces with a console that may be used to place extra Class I/II traffic on trunk links to facilitate busy hour traffic studies.

The Class I processor controls all synchronous subscriber servicing. A special interrupt driver bus connects all Class I subscriber serial line interfaces together and to the separate double buffered Class I data memory. Special hardware performs subscriber byte transfers and signaling/supervision under the control of the Class I processor. This processor also controls a Class I data DMA operation from link memories into Class I access memory.

3.2.2 Software Requirements

The functional software elements required by the system architecture shown in Figure 3-1 are not unlike those software functions, proposed in the Phase I report, for processing of link traffic. While the functions to be executed remain the same, perhaps the most important functional difference is that the real-time hardware trunk interface is controlled by the input and output processors in a slightly different manner to take advantage of the lower transmission speed and simplify the control interface.

Class I buffer addresses will be provided, along with the Class I field byte count as in Phase I but the Class II linked list block addresses will only be provided one at a time, as they are needed by input hardware. This will allow simple validation of usage by the link processors while eliminating additional address transfers over the control interface.

The software partitioning allocates the system functions over three major areas, i.e., (1) the link port processors, (2) the access port processors, which in this case are comprised of two separate elements due to real-time access constraints, and (3) the nodal processor.

3.2.2.1 Link Processor Functions

Input Processor

a. Initialization and initial Class II link list allocation (Data)

b. CCIS analysis and generation

c. Routing

d. Control switching

e. List management (input map, ack, sequence number, free list)

f. Test/diagnostic

g. Statistics.

Output Processor

a. Initialization and Class II link list allocation (Program)

b. Class I list processing (Program)

c. Class II list and queue processing (Program)

d. Test/diagnostic

e. Statistics.

The above processors operate in a background/foreground mode. Interrupt service routines will service real-time line control interface requests in addition to internal messages via packet and control FIFO bus interfaces. The communications protocol employed by these processors has been chosen to be that of ADCCP operating in a continuous ARQ-Type II mode [24]. In addition to the transmission efficiency evaluations presented in Section 6 of this report, the selection of Type II mode (retransmission of erroneous packets only) reduces the processing and queue size requirements. This is due to the decreased occupancy time spent in the transmit queues by validly received pa kets. These packets would require retransmission in other error protocols and therefore increase occupancy time. This would reflect in an increased queue size requirement on a per link basis as well as necessitate the need to prevent the reprocessing of validly received packets upon receipt of the retransmission. Time out of acknowledgments is also preferred as the method of notification over a NACK protocol since an erroneous NACK could result in a loss of control information (CCIS packets) and Nacking of NACKs becomes a serious problem since the erroneous NACK is essentially indistinguishable from any other data error.

3.2.2.2 Nodal Processor Functions

The nodal processor is functionally similar to that proposed in the Phase I study. The functions assigned to this element include:

a. I/O console function (TTY)

b. Initialization

c. Link and node administration

d. Routing/traffic updates

e. Test/diagonostic - link status

f. Node statistics/measurements

g. Debugging aids.

The most significant functional change for this processor from the Phase I study is the addition of experimental measurements and the conceptual dedication of the hardcopy I/O device for this purpose. For the model, this necessitates the inclusion of a second I/O device on the access processor to allow the nodal processor console interface to realize near full-time availability for experimental report generation, monitoring functions and parameter generation. The environment of the nodal processor is well suited to permitting this additional processing burden due to its relative insulation from the real-time link and access processes and the "quasi-background" mode in which most of its processing is performed. Additionally, as the need for new system measurements is periodically recognized, the statistics program will no doubt be enhanced and will therefore require easy access/modification coupled with the necessity to be reloaded without interruption or with minimal interference to the on-going processes.

As described in paragraph 4.3, these capabilities have already been assigned to the nodal processor due to the requirements of providing adequate software development capabilities. Continuing advantage can be taken of the up-to-date storage techniques, such as the floppy disk, and of program load capabilities for program enhancement.

3.2.2.3 Access Processor Functions

Due to the continuous and rapid digital bit stream serviced by the subscriber interface, (bi-directional device rates up to 50 Kbps) it was necessary to consider physical partitioning into two separate processors similar to link processing. Early investigations of a serial configuration introduced inter-communications complexity and system contention in accessing a common memory. The proposed design offers the functional separation shown in the architecture of Figure 3-1 and is based on the separation of data from its requisite control, for Class I processing. This requires separate Class I and Class II processors, as shown, with each processor given a method of ingress and egress to the common nodal data bus via the separate Class I and Class II memory structure and DMA controller. The details of this hardware are described elsewhere in this study.

The resultant software structure controls the real-time flow of the digitized data, which it never directly "sees", by manipulation of the required control for that data in the form of buffer addressing. Real-time hardware data-byte processing permits the Class I processor to manage local subscriber calls in a manner similar to on-net calls. Additionally, independent "class" processing permits simultaneously with Class I, the processing of all Class II data terminals attached to the Class II processor. This degree of simultaneity appears essential to avoid the potential congestion represented by multiple links addressing a single node. Due to subscriber access interfacing, parallel processing of bi-directional input and output, as found at the link level, is difficult and costly. Processing relief is given instead, at the access level, by parallel processing of Class I and Class II data with a substantial amount of real-time hardware for Class I processing.

3.2.2.3.1 Class I Processor - The functions performed by this processor include the following:

a. Class I Buffer Management - Classmark processing and subscriber buffer allocation based on 10 msecs

b. Routing - Outgoing link selection via routing table

c. Class I DMA Control - Class I buffer address transfers by DMA from various port data memories

d. Local subscriber supervision/signaling scan and control

e. Local Subscriber Switching - Address manipulation of Class I memory to create a cross-connected bi-directional data flow.

3.2.2.3.2 Class II Processor - The functions performed by this processor include the following:

a. Class II Buffer Management - Similar to Phase I, 30 word blocks will be managed as a free list memory resource for dynamically processing generated outgoing packets

b. Routing - Outgoing link selection resulting from internal processing and buffer address transfer

c. CCIS Generation/Analysis - Outgoing "call initiate" message types and initiation of Class I processes for received packets

d. Packetizing and Depacketizing - May be combined with message assembly function for device specific buffers based on terminal rate. Packetizing routines will utilize Class II access memory free list buffers for packet generation

e. Class II DMA Control - Class II buffer address transfers by DMA from various port data memories

f. Class I/II "Dummy" Message Header Generation/Control - Generation of table-driven Class I and Class II simulated traffic by console request processing.

The Class II processor has separate provision for an I/O console (perhaps teletype) as discussed earlier under the nodal processor. Provision to experimentally vary the network traffic is made possible, at minimum hardware subscriber/interface cost, by inclusion of a predetermined table of data (bit) patterns. Table generated voice and data calls may be utilized to explore important system parameters, (e.g., full frame loading, traffic mix, large message transfer) and the corresponding system behavior. Since it was desired to permit full time availability of the nodal I/O console for statistics, report generation and system monitoring, a second I/O console is provided on the Class II processor for generation of experimental traffic requests.

The software elements described herein for the SENET-DAX model provide modular functional partitioning and thus avoid the complexities and disadvantages of a large system executive. This distribution extends the characteristics of configuration flexibility, graceful degradation and higher diagnostic resolution achieved in the Phase I approach to the conceptual model. When considered in concert with the hardware structure, it is expected to yield higher performance and more flexibility for a given cost than other alternative designs considered by this study.

3.2.3 Comparison with Phase I Model

The recommended model design configuration differs some from the Phase I study results. A central difference is the absence of the nodal overflow memory and the resulting third level bus. These elements which were used in the Phase I system to store unusually high busy hour traffic, were not incorporated because the study model maximum traffic level has been considered in the sizing of the link data memories.

For laboratory implementation of the current design, single thread elements will not be made redundant. The elements concerned are the nodal processor, the two level bus structure, the system clock, and routing tables. Before the system can be removed from the laboratory environment to an operational environment, redundant backup for the single thread elements will have to be considered.

An elapsed time clock has been added to allow processing elements to measure the time required for various traffic and control procedures. Procedural measurements are given to the nodal processor for statistical correlation and presentation at the system console.

At the access level, the experimental design separates the synchronous Class I subscribers from the asynchronous Class II subscribers with a microcomputer serving each class. This differs from the Phase I recommendation of an approach that suggested the use of a low-cost microprocessor for each termination. Additional development effort would be required to supplement the Phase I study recommendation in the most economical manner.

3.3 BASIS OF CHOICE FOR THE DESIGN SELECTION

3.3.1 Design Considerations

The approach taken in choosing the distributed system architecture was intentionally iterative. Initial and repeated attemps were made to place a maximum burden on operational software. These efforts to minimize special hardware designs and produce a universally versatile programmed system were often frustrated by the complexity of the many input/output protocols required by the dynamic SENET-DAX concept. As a result, the selected system design uses some specialized input/output hardware with supervisory processor control.

The speed and complexity of a single full-duplex trunk link presents a significant challenge to any processor. A full-duplex trunk link at 230.4 Kb/s requires an input or output byte transfer approximately every 17.5 μsec. This real-time operational constraint is compounded by a multi-faceted link protocol. A complex start-of-frame (SOF) flag detection algorithm requires a bit time completion, while SOF production allows byte timing and the involved SOF correlation procedure may be performed during the 10 millisecond frame period.

The Class II portion of the frame field represents a similar challenge to a processor. An output transmission sequence contiguously includes a Class II flag, packet destination address, sequential transmission packet number, and packet text. Following the transmission of the leading Class II flag the processor must develop on a bit-by-bit basis, via a complex algorithm, a 16-bit Cyclic

Redundancy Code Check (CRCC) error-checking word. The inverte CRCC is appended to the packet text and followed by a closing Class II flag. Concurrent with these activities, the processor must search all data between Class II flags to identify any grouping of five contiguous "1" bits after which a "0" bit must be inserted. This "bit stuffing" eliminates the possible appearance of a false Class II flag (logically, 01111110) during a packet transmission.

A received Class II portion of the frame field represents a greater challenge to a processor. Detecting the Class II flag, which identifies the starting bit of the bit stuffed packet, requires a multi-step algorithm execution during a bit time. After detecting the Class II flag the processor must destuff the serial stream while simultaneously developing a CRCC. If the CRCC word is all zeros during the same bit time that the packet ending Class II flag is detected (parallel algorithm), the received packet is considered valid.

During the performance of the link protocol procedures, the processor must also perform many other non-trivial tasks. Incessant link input/output byte transfers to and from memory are directed by tables containing the starting address and byte count length of each packet. These tables require organizing and constant modifying attention. The generation and analysis of Class II packets with the associated routing procedures consume much processing time. The following section contains a detailed presentation of the background processing routines.

It is quickly recognized that special input/output hardware could perform many of the severe link interfacing timing requirements more efficiently than a processor. Placing DMA mechanisms, SOF and Class II flag control, CRCC circuits and bit stuffing into the link input/output hardware allows the link processor to operate in a supervisory mode.

Even after removing the severe bit time operational demands from the immediate control of the link processor, a significant processing load exists. Initial attempts to partition trunk link processor tasking to a link controller resulted in a dual processor architecture where one processor (link controller) closely controlled both input and output DMA byte transfers while managing output packet sequence number lists. The second processor (link processor) was also assigned input/output functions such as packet generation and analysis, Class I connection map management, routing and control switching over the control bus, link memory allocation, and output queueing. This functional partitioning exposed three areas of difficulty. The successive byte incremented DMA block transfer operations for both input and output link data consumed a great amount of processor time, leaving little time for packet sequence number lists and the necessary communications with the link processor. Interprocessor communication between the link controller and link processor represented a major limiting factor in the operational viability of this dual processor link port architecture. The amount of interprocessor data necessary and the slow rate of transfer available would have resulted in link data input/output delays.

Finally, the large processing load placed on the link processor would have resulted in an impossible processing load.

As shown in Figure 3-2 and the following paragraph on software timing, the recommended design configuration separates link input and output functions for control by individual microcomputers. This functional partitioning minimizes link input and output processor intercommunications, still utilizes a high percentage of processor time, but represents the most efficient, cost-effective, and viable option. This functional arrangement closely resembles the processor architecture recommended in Phase I.

At the access port a dual processor architecture is also required. The functional partitioning separates Class I and Class II tasks with their respective synchronous and asynchronous terminals. Similar to that described above for link port processing, an evolutionary process which initially utilized an access controller and access processor concluded with the Class I and Class II processors. The subscriber terminal speed buffering with the multiplicity of Class I subscriber type signaling/supervision and rate servicing requirements, presented an impossible task for direct processor servicing. Therefore, special input/output hardware directly controls the Class I subscribers under the control of the Class I processor. In view of the small quantity and low asynchronous rate of anticipated Class II subscribers, the access port can be interfaced to the nodal bus directly thrugh a Class II processor.

3.3.2 Timing Analysis

3.3.2.1 General

The timing analysis for the input processor operational program has taken into account both the time critical requirements of input processing, and the need for a secondary processor scan. This has resulted in a link input processor structure which is partitioned into a real-time (foreground) and scan (background) mode. All real-time processing will be under the control of the Real-Time Interrupt program. Processing of the interrupt will be kept to a minimum, with input information stored in data buffers for later background scan processing. All tasks that can be delayed or performed on a background scan basis are scanned by the Control Scheduler program. It is this program which will cycle through each of the background scan subprograms allowing individual task processing to be done.

The two major processing areas (interrupt servicing and control scheduling) in the link input processor operational software are discussed below. The interrupt driven program structure permits more efficiency and minimizes the program loading on the processor due to software scanning.

3.3.2.1.1 Real-Time Clock Interrupt - A real-time clock function has been assumed for purpose of this analysis, with this function initiating control address transfers for storage of input voice and data. The clock interrupt rate is 1 msec which approximates the time required to store an incoming packet.

The large number of interrupts/sec (determined on a 1 ms basis to be 1000 interrupts/sec) underlies the need for efficient and compact software coding of these routines.

3.3.2.1.2 Control Scheduler Program - All background processing for the Line Input Processor will be under the direct command of the Control Scheduler. Its function is to sequentially scan through its task queue polling each subprogram for activity. When activity is requested, the control scheduler will call the responsible subprogram for processing.

The following task subprograms will be called by the Contol Scheduler during a background scan and form the basis for the following timing estimates:

a. Free memory-list management

b. Packet analysis

c. Transmit packet control

d. Ack/No Ack processing

e. Maintenance

f. Statistics.

3.3.2.2 Link Input Processor Timing

The timing analysis has been performed for the sub task elements of the control scheduler in the Link Input Processor. The assumptions which were made for the timing analysis are itemized in Appendix I.

All calculations are also shown in Appendix I and are summarized in Table 3-1 contained in this section.

The basic plan in this analysis was to identify the time critical processor requirements in the Link Input processing. Hence the operational software is broken into a real-time processing element and a control scheduler element; the purpose being to minimize the timing on the interrupt routine. The important loading functions are discussed below in order of time dependence.

3.3.2.2.1 Real-Time Interrupt Timing - Packet storage requirements have indicated that a 1 ms interrupt cycle will be used. This means that there will be 1000 interrupts/second which must be processed. Analysis has shown that a 50 instruction interrupt handling routine should be considered. The processing loading as figured on a 1000 ms (1 second) capacity is calculated to be 160 ms or 16 percent processor loading. The high number of interrupts/second shows the need to minimize the size of the interrupt processing routine.

3.3.2.2.2 Control Scheduler Timing - Scan requirement estimates show that a 4 ms background scan is required to set up the processing of operational software background routines. An analysis of the maximum number of executed instructions on a loop scan found that 100 instructions would be required. This results in a 78 ms execution time or 7.8 percent link input processor loading.

3.3.2.2.3 Free Memory Background List Management - The timing impact of list management has been considered using the maximum number of addresses which may be processed in a given scan. Using the absolute

TABLE 3-1. LINK INPUT PROCESSOR TIMING SUMMARY

	LOADING @ 500 PACKETS/SEC	LOADING @ 200 PACKETS/SEC
● REAL-TIME CLOCK ROUTINE	16.0%	16.0%
● CONTROL SCHEDULER	7.8%	7.8%
● FREE MEMORY — LIST MGT.	2.2%	.9%
● PACKET ANALYSIS	30.0%	12.0%
● TRANSLATION	4.5%	1.8%
● ROUTING	4.2%	1.7%
● CCIS PACKET GENERATION	.03%	.03%
● TRANSMIT PACKET CONTROL	6.0%	2.4%
● ACKNOWLEDGE/NO ACKNOWLEDGE	7.5%	3.0%
● MAINTENANCE & STATISTICS	.6%	.6%
TOTAL PROCESSOR LOADING	78.8%	46.2%
PROGRAM RESERVE	21.2%	25.0%

8083-76E

maximum of 500 input packets/link, the average number of addresses required would be 250 transfer addresses. The routine is anticipated to consist of an overhead portion of sequence setup followed by a loop list process. The execution of free list processing results in at most a 22.5 ms or 2.25 percent loading of the processor which is not significant. The speed of the individual scan (that is the number of instructions) must be kept low so as to minimize processor loading.

3.3.2.2.4 Packet Analysis Timing - For this calculation a figure of 500 packets/sec was considered an absolute maximum of the Class II traffic per link. The longest potential processing loop of the analysis routine has been estimated to be 200 instructions. The result of this processing is a 300 msec processor time or a 30 percent loading on the processor. This large loading indicates the importance of this aspect of the processing and careful attention should be paid to its organization in the final design.

3.3.2.2.5 Translation and Routing Timing - Although these routines are called as subroutines and are not directly involved in the background processor scan, their timing impact should be measured when taking into consideration the total link input processor loading. We consider that all data packets will be translated and routed prior to control switching to another link. The calculation in the table shows that the total impact will be less than 10 percent loading.

3.3.2.2.6 Transmit Packet Control Timing - After a CCIS or data packet has been set up and formatted, control information for the packet must be transmitted to the output link processor. In the worst case, a maximum of 500 packets/sec will have to have control information transferred. The subroutine to do this is estimated to be 40 instructions in length, but the impact of 300 packets/sec results in a 6 percent loading just for one task.

3.3.2.2.7 Acknowledge/No Acknowledge Timing - Timing for the absolute maximum case considers that 500 packets/sec will have to be acknowledged when they are processed by the Link Input processor. This timing analysis does not consider the timeout monitoring that is taking place whenever a packet is transmitted from the output processor. This will be examined in the timing analysis of the Link Output Processor. The processor loading on the basis of a maximum of 500 packets (per link) is determined to be 7.5 percent.

3.3.2.2.8 Maintenance and Statistics - This routine is the lowest level of time importance but the great number of instructions anticipated for the function indicate that there is a loading effect on the processor. On the basis of estimated requirements it has been assumed that 200 instructions will be executed in the worst background maintenance and statistics case. This is equivalent to a 6 percent loading figure.

3.3.2.2.9 Link Input Processor Timing Conclusions - Processor loading for an absolute maximum per link of 500 packets/second, and .13 CCIS calls/sec is calculated to be 78.8 percent. This seems like high loading but it must be emphasized that the peak traffic/link condition is assumed. For an average busy hour condition of 200 packets per link, the processor loading drops down to 46 percent. Table 3-1 is a summary of the calculations of these traffic conditions.

3.3.2.3 Link Output Processor

Similar to the Link Input Processor, the design of the LOP considers the time critical sequences of processing the Class I and Class II messages. Thus the link output structure is partitioned in a real-time scan and background scan mode. All real-time processing is considered to be under the control of the Real-Time Interrupt program for purpose of this analysis. The technique and requirements used for the Real-Time Clock routine will be the same as previously discussed for that of the Link Input Processor.

All tasks that can be performed on a background scan basis are scanned by a Control Scheduler program. Each of the two major processing areas in the LOP software are discussed below.

3.3.2.3.1 Real-Time Clock Interrupt - This routine is entered via a 1ms clock interrupt. Comparable to the interrupt processing for the Link Input Processor, the service routines process the requisite control information for the real-time transfers from the link data memory.

3.3.2.3.2 Control Scheduler Program - Like the input processor, all background processing for the LOP will be under the direct command of the Control Scheduler. The function of the Control Scheduler is to scan sequentially through its task queue polling each subprogram for activity. When activity is requested, the Control Scheduler will call the responsible subprogram for processing.

The following task subprograms will be called by the Control Scheduler during a background scan and form the basis for the following timing estimates.

a. Free memory-list management

b. Background queue entry/transmit queue processing

c. Timeout processing

d. Output address generation

e. Process delete channel message

f. Maintenance statistics.

3.3.2.3.3 Link Output Processor Timing - The timing analysis has been performed for the time critical elements of the Link Output Processor. All calculations are shown in Table 3-2. A number of timing techniques common to both the input and output routines have not been re-included in this paragraph.

3.3.2.3.4 Free Memory Backgrund List Management - Refer to paragraph 3.3.2.2.3 for discussion of list management timing requirements.

3.3.2.3.5 Background Queue Entry/Transmit Queue Processing - For this calculation a figure of 500 packets/sec was considered as an absolute maximum for the Class II traffic per link. It has been determined that the routine to process the allocation of the incoming packets to

TABLE 3-2. LINK OUTPUT PROCESSOR TIMING SUMMARY

	LOADING @ 500 PACKETS/SEC	LOADING @ 200 PACKETS/SEC
● REAL-TIME CLOCK ROUTINE	16.0%	16.0%
● CONTROL SCHEDULER	7.8%	7.8%
● FREE MEMORY – LIST MGT.	2.2%	.9%
● BACKGROUND QUEUE PROCESSING	12.0%	4.8%
● TIMEOUT PROCESSING	.1%	.1%
● OUTPUT ADDRESS GENERATOR	9.0%	3.6%
● MAINTENANCE STATISTICS	.6%	.6%
TOTAL PROCESSOR LOADING	47.7%	33.8%
RESERVE (ASSUMED)	25.0%	25.0%

8081-76E

the appropriate queue will take an estimated maximum of 60 instructions. This results in a software impact of 90 ms on the output processor or a 9 percent processor loading.

3.3.2.3.6 Timeout Processing - The impact of this routine is dependent on the number of packets per second which are received incorrectly and allowed to time out. This number is dependent on the particular burst error rate (BER) of the transmission media: i.e., for a BER of 10^{-5} and a packet size of 224 bits, only 1.1 packets/sec are in error, whereas for a BER of 10^{-3}, at least 100 packets/sec are in error. However, due to the small number of instructions to process a timeout, the processor loading of 2 percent at maximum is not a significant factor.

3.3.2.3.7 Output Address Generator - The maximum link constraint of 500 packets per second results in a 6 percent loading impact due to the fact that this routine is called repetitively.

3.3.2.3.8 Maintenance and Statistics - Refer to the discussion in paragraph 3.3.2.2.8 on Maintenance Statistics.

3.3.2.3.9 Link Output Processor Timing Conclusions - Processor loading for an absolute maximum per link of 500 packets/second and .13 CCIS calls/sec is calculated to be 47.7 percent. For an average condition of 200 packets per link, the processor loading decreases to 33.8 percent. Table 3-2 is a summary of the calculations of these traffic conditions.

3.3.2.4 Summary of Timing

A timing analysis of the Link Input Processor and the Link Output Processor has been presented. The figures are herein summarized for the peak and average busy hour cases below. The reserve requirement for future program expansion was set at 25 percent.

TABLE 3-3. TIMING SUMMARY: LINK INPUT PROCESSOR AND LINK OUTPUT PROCESSOR

	PEAK TIMING 500 PACKETS/SEC PERCENT	AVERAGE 200 PACKETS/SEC PERCENT
Link Input Processor	78.8	46.2
LIP Reserve	25.0*	25.0
Link Output Processor	47.7	33.8
LOP Reserve	25.0	25.0

*It should be noted that the Link Input Processor loading does not allow a 25 percent reserve requirement. The actual forced reserve requirement is 21.2 percent.

The above information is useful in determining the impact of various architectures on the DAX timing. In particular, a discussion of the Central Processor approach is evaluated below.

The basic characteristic of the C. P. approach is the combination of the Link Input Processor and the Link Output Processor. In addition an access switch will be integrated into the C. P. requirements. Combination of the LIP and LOP reduces software duplication, but on the other hand the traffic peak handled by the processor, and its average offered traffic, increases greatly.

3.3.2.5 Central Processor Impact on Duplication

It is noted that there are a number of routines which can be considered common to both input and output processing. These include the following:

a.	Real-Time Clock Interrupt Routine	8.0 percent
b.	Control Scheduler	3.9 percent
c.	Free Memory/List Management	2.2 percent
d.	Maintenance and Statistics	6.0 percent
	Total	20.1 percent

In addition the transmit Packet Control (with 6.0 percent loading) in the Link Input Processor will not be necessary since the same processor does Input and Output: therefore, no interface routine is necessary. Hence the total impact of the Central Processor Approach is a total decrease of 26 percent.

The combined impact timing is considered in Table 3-4. It is evident that with a peak packet load the requirements of loading on the processor exceed its capacity. Even with an average busy hour packet figure of 200 packets/sec, the loading using the combined effects jumps 18 to 31 percent over the distributed approach for the 3-link model under consideration.

It must also be kept in mind that the summary timing of Table 3-4 does not include estimates for the interrupt processing required by the access subscribers. One method of evaluating the viability of the central processing approach is to consider the impact of this

TABLE 3-4. CENTRAL PROCESSOR (LIP & LOP) TIMING SUMMARY

	LOADING @ 500 PACKETS/SEC	LOADING @ 200 PACKETS/SEC
• REAL-TIME CLOCK ROUTINE	24.0%	24.0%
• CONTROL SCHEDULER	11.7%	11.7%
• FREE MEMORY LIST MGT.	2.2%	.9%
• PACKET ANALYSIS	30.0%	12.0%
• TRANSLATION	4.5%	1.8%
• ROUTING	4.2%	1.7%
• CCIS PACKET GEN.	.03%	.03%
• BACKGROUND QUEUE PROC.	12.0%	4.8%
• TIMEOUT PROCESSING	.1%	.1%
• ACKNOWLEDGE/NO ACKNOWLEDGE	7.5%	3.0%
• OUTPUT ADDRESS GEN.	9.0%	3.6%
• MAINTENANCE & STATISTICS	.6%	.6%
TOTAL	105.83%	64.2%
PROGRAM RESERVED	—	25.0%

8082-76E

access processing load. An interrupt service routine must complete servicing of all subscribers in less than 160 μsec since this is the byte transfer rate for <u>one</u> KY3-50 kilobit line. Since the service time for up to 12 subscribers would require about 90 μsec to check and store all input bytes, only 70 μsecs or 44 percent of the total processor time would be available for processing. Reference to Table 3-4 indicates that 64.2 percent is required to be available for average busy hour processing and thus the single processor would be overloaded by addition of the access subscriber scanning function.

3.4 DETAILED HARDWARE DESIGN FOR COST ESTIMATES

The detailed hardware designs shown in Figures 3-3 and 3-4 provide a firm basis in estimating hardware costs. Each processor is shown with program memory and a peripheral input/output bus. Attached to the peripheral buses are parallel line interface (PLI) and serial line interface (SLI) modules. An SLI provides an asynchronous interface from a processor to a teletype or other data terminal device. A PLI allows a multi-bit parallel input or output with peripheral handshaking, strobed by an appropriate nodal clock phase.

All port data memories, which together comprise the total nodal memory on one data bus, are surrounded by clock selected address and data gating circuits. Incoming data from a link or access terminal is given X clock phases for data storage, the data bus is allowed at least X clock phases for data extraction, and the appropriate memory management port processor is provided X clock phases for read or write operations. A comparator is associated with every port memory to monitor the upper three bits on the address portion of the data bus. When an address appears on the bus which requires data from a particular port memory, the port comparator enables tri-state bus driving buffers to transmit the requested data.

An elapsed time clock is given a periodic clock phase to propagate incremented a 16-bit time refere. onto a portion of the 28-bit data section of the control bus. .hen a processor is commanded to statistically analyze a control operation, a snap-shot reading of this clock at the initiation and conclusion of the operation provides

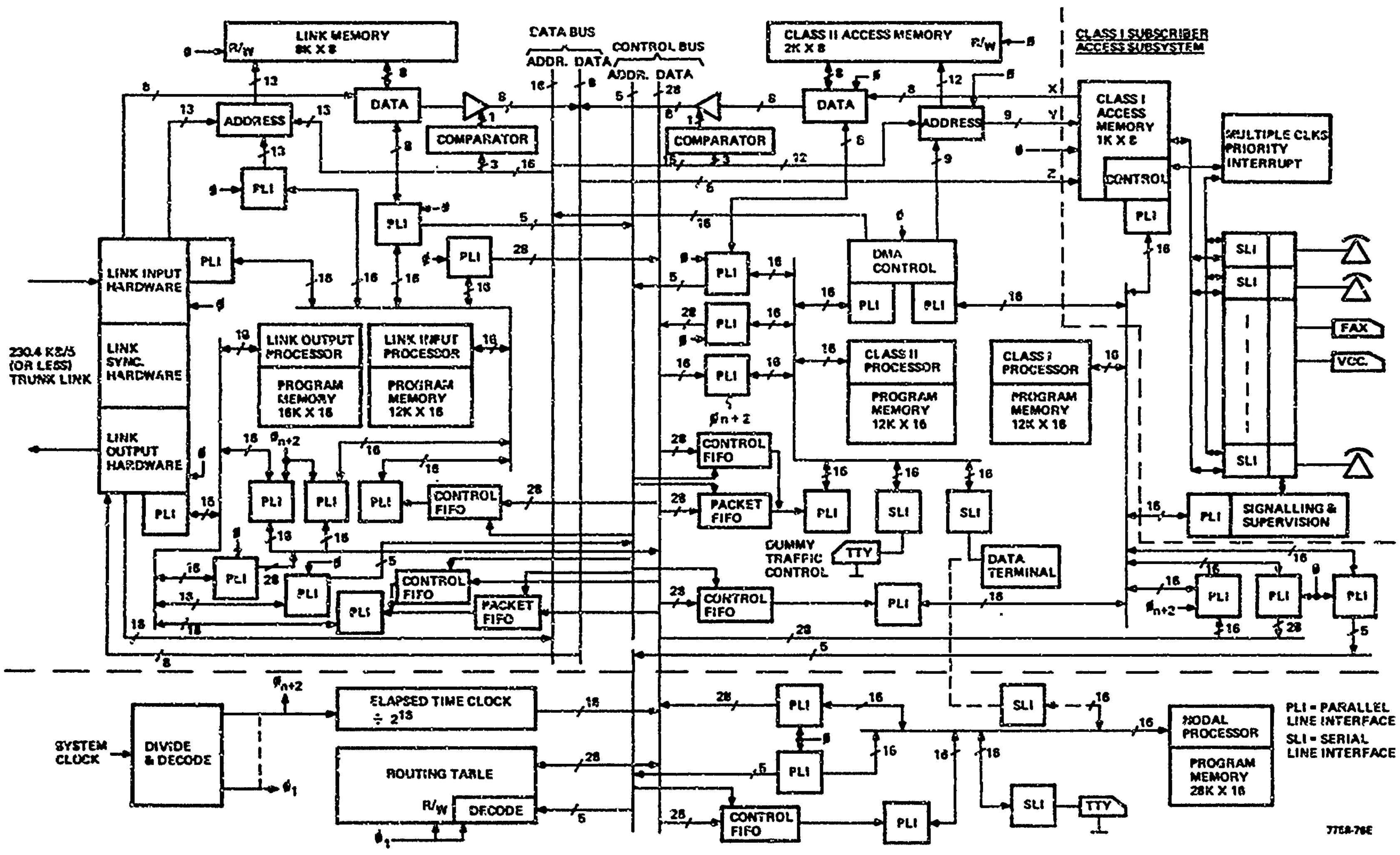


Figure 3-3. SENET-DAX Model Functional Hardware

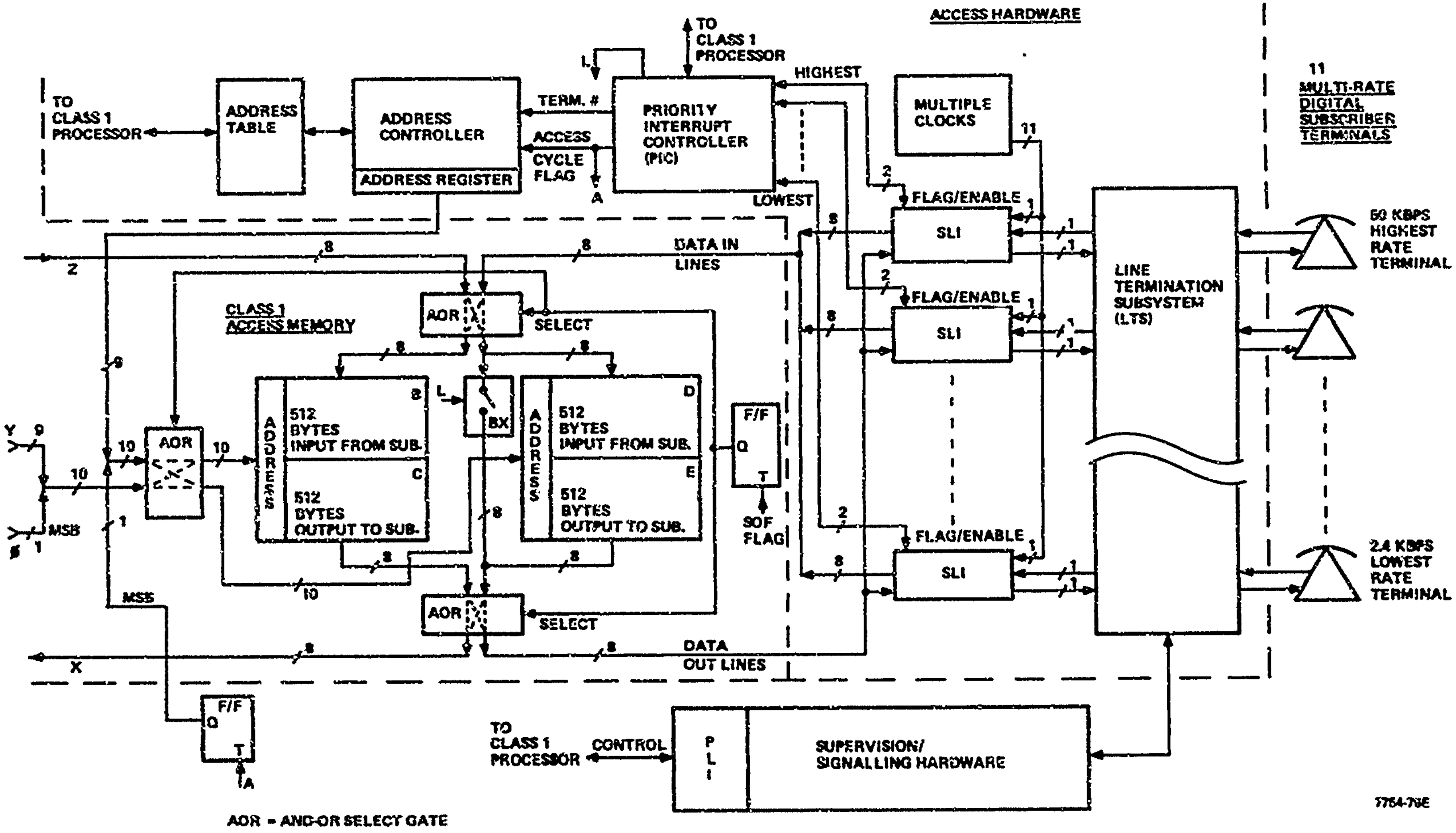


Figure 3-4. Class I Subscriber Access Subsystem

the relative information needed to quantize the operational time consumed. Based on the maximum trunk link rate of 230.4 Kb/s, the elapsed time clock resolution is about 34.7 sec with a maximum window time of approximately 2.27 seconds.

The routing table is updated periodically only by the nodal processor, gated by the nodal clock phase 1. Therefore, the routing table read/write line is also controlled by phase 1 to allow a read or write during that phase and only a read operation at all other times.

Previous sections of this report explain nodal processor functions, link port hardware and processing operations, and access port processing with Class II data. The following paragraph describes the hardware operations in the Class I subscriber access subsystem.

3.4.1 Class I Subscriber Access Subsystem

The purpose of this paragraph is to describe the operation of the SENET-DAX Class I Subscriber Access Subsystem. First, access requirements will be determined for this scaled down version of the Phase 1 SENET-DAX. Then the functional hardware will be described, including a byte by byte description of data transfer. Finally, the subscriber capacity of the access subsystem will be discussed.

In order to demonstrate SENET-DAX concepts, requirements for the Class I subscriber access subsystem have been set. The subsystem must be able to input digital subscribers at synchronous data rates between 2, 4 Kbps and 50 Kbps. To have a reasonable representation of subscribers in this range and to illustrate the workability of the

SENET-DAX concept, the following typical distribution of subscriber terminals has been considered.

Quantity	Terminal Type	Data Rate
1	KY-3	50 Kbps
4	CVSD VOICE	32 Kbps
4	CVSD VOICE	16 Kbps
1	FACSIMILE	9.6 Kbps
1	VOCODER	2.3 Kbps
11 TOTAL		

The Class I subscriber access subsystem must be able to service the Class I data, supervision and signaling at the 11 typical multi-rate subscriber terminals listed above. The supervision and signaling techniques that interface with the access subsystem include dual tone multifrequency, tone-on-idle, dial-pulse and digital codewords.

To service the above requirement, a proposed Class I subscriber access subsystem is shown in Figure 3-4. The subsystem consists of Class I access memory, supervision/signaling hardware, and access hardware. The access memory is subdivided into four blocks of 512 bytes each (B,C,D,E). This block grouping is based on the maximum amount of data that can be transferred in one direction between the nodal bus and the 11 suscriber terminals in 10 milliseconds (288 bytes rounded up to the most significant bit). The access memory is also grouped into two 1024-byte blocks. Each of these blocks is alternately accessed from either the nodal data bus (by Ø, X, Y and Z) or the access hardware. The start-of-frame (SOF) flag toggles a flip-flop which gates the address and data lines of the access hardware or nodal bus to an alternate 1024-byte block of Class I access memory every 10 milliseconds. Assume for the moment that the access hardware has access to memories D and E. The line termination subsystem (LTS) interfaces the digital subscribers with the serial line interfaces (SLI) and the supervision/signaling hardware. The LTS separates data from signaling/supervision for each subscriber terminal. The supervision/signaling hardware interfaces with the Class I processor for control.

Once signaling and supervision are processed and a connection is made between two subscribers, the LTS gates Class I data from a subscriber terminal to a corresponding SLI. The SLI is a full-duplex synchronous serial-to-parallel converter that is supplied with a clock rate that is the same as the terminal rate which it services. The SLI is double buffered to hold an incoming byte for 1 byte time. Upon receiving eight bits of serial data the SLI generates a flag. The flag from each SLI is presented to a priority interrupt controller (PIC).

The subscriber data rate at which a SLI operates determines the servicing priority assigned to its SLI flag. Servicing priority is assigned as follows:

Data Rate	Priority	
50 Kbps	1	highest
32 Kbps	2	
32 Kbps	3	
32 Kbps	4	
32 Kbps	5	
16 Kbps	6	
16 Kbps	7	
16 Kbps	8	
16 Kbps	9	
9.6 Kbps	10	
2.4 Kbps	11	lowest

The PIC acknowledges the highest priority interrupt and masks out all lower priority interrupts until the highest priority interrupt is serviced.

The priority interrupt results in a complete Class I access memory cycle. Initially the address table contains a starting address and frame time byte count for each subscriber. The byte count for each terminal is the number of bytes of storage that are allocated for 10 milliseconds of subscriber data. The starting address is assigned to allocate a non-overlapping memory block for each subscriber. Along with these quantities, a current byte count for each subscriber is kept in the address table and updated after each access cycle.

A Class I access memory cycle proceeds as follows. The PIC first sends the subscriber terminal number to the address controller. The address controller looks up the starting address and current byte count. The address register is loaded with the current address which equals the starting address plus the current byte count, and points to a location in a 512-byte input block of memory (D). The PIC then enables the data-in lines of the SLI and stores the incoming data byte in memory D at the location pointed to by the address register. The access cycle then complements the flip-flop MSB to point to a location in output memory E from which a byte of output data is transferred to the interrupting SLI. This byte is serially clocked to the subscriber terminal. The preceding process constitutes one access cycle. The current byte count is now updated, allowing the next priority interrupting SLI to be serviced.

Ten milliseconds from the first access cycle, the current byte count will equal the final byte count for all serviced subscribers. When this occurs a SOF flag will interchange the Class I access memory address lines between memory blocks (B,C) and (D,E). While the access hardware is performing access cycles on memories B and C, any link output controller, via the nodal bus, is allowed to perform block burst transfers of data from B. Simultaneously, the Class I processor controls similar DMA transfers into memory C from any link memory. Discrete burst transfer rates can be as high as the maximum link data rate.

The above process is used exclusively for network calls. For the interconnection of compatible local subscribers, at the same port, subscriber data does not use access memory. The SLI's for the local subscribers are connected by the PIC using the bus switch (BX) during the priority interrupt cycle.

SECTION 4

EQUIPMENT/SOFTWARE SURVEY INVESTIGATION

4.1 MINICOMPUTERS VS MICROCOMPUTERS

During the evaluation of design alternatives, a number of configurations were considered, including those calling for higher processing capability due to the centralization (and therefore serialization) of processing. An early outgrowth of these considerations was the realization that a centralized approach, utilizing minicomputers exhibited difficulties (such as timing and memory access constraints, see paragraphs 2.4 and 2.6) that were not offset by raw speed, floating point options, comprehensive libraries and large executive programs for resource control. Further examination also led us to conclude that existing applications software, offered by manufacturers, could not be directly applied without extensive and difficult modifications. The main survey effort was therefore focused on the microcomputer field and the available development tools required by a microcomputer based system.

4.1.1 Microcomputers

This decade has seen a rapid proliferation of microcomputer manufacturers within the semiconductor industry, utilizing primarily a metal-oxide-semiconductor (MOS) technology. The ability to choose an optimal processor for a specific application is complicated by not only the number of products but also by similar claims of capability, sparse application notes and the general unavailability of user evaluations due to the infancy of this new market. At this writing, 4, 8, 12, and 16-bit processors have been developed with the 8-bit processor currently dominating the market. A typical microprocessor-on-a-chip architecture is shown in Figure 4-1. The on-chip computational capability, aside from microcode, is of course determined by the number of registers, their width and their connectivity within the chips. Figure 4-1 shows a typical address and data bus arrangement for accessing memory. The 16-bit address bus gives an addressing capability of 64k but the 8-bit data bus requires multiple accesses for fetching 16-bit operands or address retrieval. These architectural limitations impose a careful matching of processor-to-application for efficient problem processing and require an examination of the software development tools that may be provided, since these tools substantially affect the development time, cost, and reliability of the product.

4.1.2 Software Characteristics

In order to enhance their capability, certain manufacturers have produced microcomputers, replicating the instruction set of existing minicomputer product lines. This degree of compatability allows much of the sophisticated minicomputer support software to be made available to support the development effort.

Other vendors have attempted to provide their own basic assembler programs, both cross and resident, but by minicomputer standards these are not extensive.

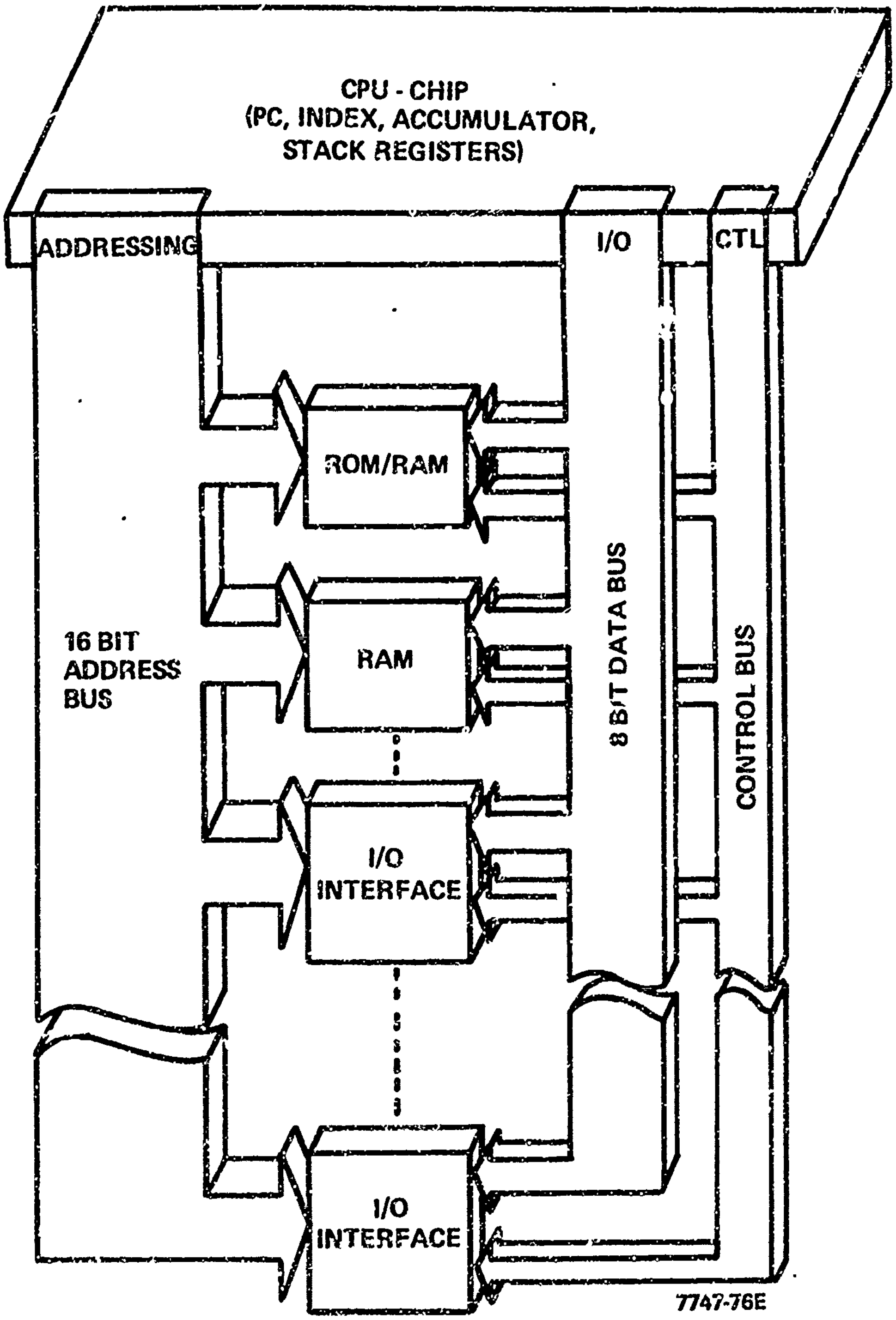


Figure 4-1. Typical Microprocessor Component Architecture

The development of a resident language processor is a non-trivial task and it is not likely that high quality language processors will be produced by volume-oriented semiconductor vendors due to the large required investment. This is readily apparent in the existing software now being offered by most semiconductor microcomputer manufacturers. Most cover only basic translation support (language processor) and some form a debug capability (monitor). In many cases, little or no support is provided for design, code and loading of programs.

According to Ogdin, [14], monitors are notoriously poor and oversized. Since size is an important measure of monitor quality, his rewriting efforts have realized an increase of over 50 percent in monitor space. Assemblers also appear to be of generally poor quality (resident), oversized and error prone.

Tables 4-1, 4-2 and 4-3 summarize three types of resident assembly language programs available today for symbolic development of programs: Table 4-1, Assembly Language; Table 4-2, Macro Assembly Languages; and Table 4-3, System Implementation Languages. Basically assembly language is still the most efficient, where tight code and minimum size may be critical. Once the single assembler is left behind in favor of MAL or SIL development, less control is exercised over generated code and is somewhat less efficient. The tradeoff here is speed and increased reliability versus tightness and efficiency of code.

TABLE 4-1. ASSEMBLERS

VENDOR	μ-PROCESSOR	RESIDENT	CROSS	RELOCATABLE
DATA GENERAL	μ-NOVA	YES	YES	YES
DIGITAL EQUIP	LSI – 11	YES	YES	YES
FAIRCHILD	F8	YES	YES	PLANNED
INTEL	4004	INTELLEC	YES	NO
	4040	INTELLEC	YES	NO
	8008	INTELLEC	YES	NO
	8080	INTELLEC	YES	NO
MICRO	F8	NO	YES	NO
	M6800	NO	YES	NO
	4004	NO	YES	NO
	4040	NO	YES	NO
	8008	NO	YES	NO
	8080	YES	NO	NO
MOSTEK	F8	PLANNED	YES	NO
MOTOROLA	M6800	EXORCISER	YES	NO
MITS	8080	ALTAIR 8800	NO	NO
NAT'L. SEMI	PACE	YES	YES	YES
NEC	μ-COM 8	YES	YES	NO
RCA	COSMAC	YES	YES	NO
SIGNETICS	2650	PLANNED	YES	PLANNED
TEXAS INSTR.	TMS9900	NO	PLANNED	NO

7749-76E

TABLE 4-2. MACRO ASSEMBLERS

VENDOR	μ-PROCESSOR	RESIDENT	CROSS	RELOCATABLE
DATA GENERAL	μ-NOVA	YES	YES	YES
DIGITAL EQUIP.	LSI – 11	YES	MACRO – 11	YES
FAIRCHILD	F8	PLANNED	YES	PLANNED
INTEL	4004	NO	YES	NO
	4040	NO	YES	NO
	8008	NO	YES	NO
	8080	NO	YES	NO
MICRO COMPUTER ASSOCIATES	MCS6500	JOLT	NO	YES
MOTOROLA	M6800	PLANNED	YES	NO
NEC	μ-COM 8	NO	YES	NO
RCA	COSMAC	YES	PLANNED	NO
SIGNETICS	2650	PLANNED	YES	PLANNED
ZILOG	Z-80	YES	YES	NO

7755-76E

TABLE 4-3. SYSTEM IMPLEMENTATION LANGUAGES

VENDOR	LANGUAGE	μ-PROCESSOR	RESIDENT	CROSS	RELOCATABLE
FORTH	FORTH	LSI – 11	YES	NO	YES
		COSMAC	YES	NO	NO
INTEL	PLM	8003	NO	YES	NO
		8080	NO	YES	NO
MOTOROLA	MPL	M6800	TO BE ANNOUNCED		
NAT'L. SEMI	SM/PL	PACE	NO	YES	YES
NEC	PLM	μ-COM 8	NO	PLANNED	NO
SIGNETICS	PLP	2650	NO	YES	YES
SYCOR	PLM	8080	NO	YES	YES
ZILOG	PLZ	Z-80	PLANNED	YES	NO

7756-76E

4.1.3 Macro Assembly Language (Table 4-2)

Macro capability permits the generation of a predetermined number of machine language instructions for each MAL instruction. Capabilities generally vary from conditional selection of program segments to basic text replacement. This level represents a step to improve the reliability of the generated code, reduce its development time and still produce reasonably tight, efficient code. As such, it falls about half way between an AL and a higher level language and represents a good compromise. The Intel MAL offers only basic text replacement capability while the announced RCA MAL is expected to have both conditional and parametric capability. Motorola is understood to be developing a MAL with significant capability for the M6800 [12].

4.1.4 System Implementation Languages (Table 4-3)

Microcomputer based SIL's are still in an early development stage but are generating high industry interest. Conceptually, a SIL increases the efficiency of generated code by use of algorithmic representations that relate to and take advantage of the specific architecture of the machine. Intel's PLM is a well known example of this type. The increase in efficiency is obtained only with a loss of generality with respect to compatability with other processors. Problem areas in present SIL's include address arithmetic, resource/register allocatirn and interrupt handling, but new machine architectures may help to solve these difficulties in the future [8].

Some industry efforts are beginning to concentrate on improving offered language products which should be of future benefit to produce better development support software. Many products are still experimental in nature with others nearly primitive in scope. Early users must assist manufacturers in evaluation and are improving those versions that are available today at their own expense.

In light of the uncertain staying power or support committment of some semiconductor manufacturers, and the need for comprehensive development tools consistent with processor capability, this study effort focused on two manufacturers that provide substantial software support due to their compatible minicomputer product lines. This focus is consistent with study guidelines to utilize existing firmware/software processing modules. Both Data General's μ-Nova and Digital Equipment's LSI-11 microprocessor permit proven existing software tools to be directly applied to the development of a peripheral-limited multi-microprocessor system. These two products are more fully described in the following paragraph on Operational capability.

4.2 OPERATIONAL CAPABILITY, AVAILABILITY AND COST

One major stipulated goal for this supplemental design study was to maximally utilize existing hardware and software modules in the recommended design configuration. As discussed in the previous paragraph, this criterion prevented serious consideration of any system manufacturer that did not supply a complete hardware family, extensive software support, and off-line software development capability. In view of the above, and the limited time frame for this study, the equipment/software survey investigation selected the Digital Equipment Corporation LSI-11 and the Data General Corporation μNova microcomputer families for comparative analysis and evaluation of their applicability for the recommended design configuration.

The LSI-11 and μNova differ considerably in their central processor and microcomputer system architectural approaches. While the LSI-11 with its multi-chip central processor units (CPU) places all memory , peripheral interfaces, and other system modules on one parallel multiplexed address/data bus, the μNOVA uses a two bus architecture. The μNOVA with a single-chip CPU performs memory transfers over a parallel multiplexed address/data bus while providing a second bus structure to communicate with input/output controllers (IOC). Data and control/address information on the input/output bus is transferred at an effective $16.\bar{6}$ MHz bit rate. This is accomplished via two parallel $8.\bar{3}$ MHz bus lines each with nine serial time slots for a control bit and data byte. Ultimately, the input/output word (16-bit) transfer rate is approximately 920 nsec. To permit these high speed serial transfers in parallel, each input/output peripheral element must interface this bus through a special three-chip set available from Data General.

To assist in minimizing equipment costs and space requirements MOS dynamic RAM memories were chosen. In the LSI-11 system two methods of dynamic RAM refresh are available. The CPU may perform a total memory refresh lasting at least 130 μsec and occuring every 2.4 msec. If 130 μsec is too long for a complete halt to all CPU processing, a special bus module may be added to initiate partial refresh cycles, each lasting 2 μsec and occurring in sufficient frequency so that all memory is refreshed within 2.4 msec. The μNOVA CPU contains a hidden refresh mechanism which automatically initiates partial refresh cycles during CPU computations.

The LSI-11 utilizes microcoded CPU routines to perform the basic, but very powerful, PDP-11 instruction set. Conversely, the μNOVA uses a more discreet step-oriented instruction set which is basic to upper level Data General minicomputers. To determine which machine provides an opertional processsing advantage, future bench-mark programming will have to be performed using routines for the recommended design configuration which are characteristic of most system operations.

According to company representatives, the LSI-11 microcomputer family is available with a thirty-to-ninety day delay and the μNOVA microcomputer family will be ready for quantity shipping in January 1977. The average estimated cost for a microcomputer-based network of two, three or four nodes is $89,245.00, $159,594.00, and $228,343.00 respectively. These estimated costs include an off-line development system and are amplified in Section 7 and detailed in Appendix II.

4.3 DEVELOPMENT SYSTEM OFF-LINE

One of the most difficult aspects of implementing a microcomputer-based system is the development and debugging of resultant programs. In view of the numerous processing elements identified by this study in the proposed architecture, it was concluded that the following capabilities would be essential for an off-line development system for the SENET-DAX. Figure 4-2 presents a block diagram of the proposed configuration.

a. Remote Load - Loading the nodal processor "down line" by the development system upon generation of appropriate linked/load modules

b. Multiple Printers - Simultaneous development of software by two or three users

c. Hi-Speed Printer - Ability to print hardcopy listings within a reasonable time period

d. Off-line floppy disk storage (dual).

As configured, this system operating with Remote 11, under operating system RT-11, provides a multi-user development capability capable of producing managed assemblies with revision level capability. Dual floppy disks provide off-line storage for developed programs and complete relocatable load modules may be "automatically" loaded into the SENET-DAX nodal processor via Remote-11. Down-line loading of the nodal processor is an important capability due to the number of peripheral-limited link processors which may now be program loaded by the nodal processor upon receipt of the programs transmitted to it under Remote-11. These capabilities and the dedicated purpose of this off-line system will favorably and heavily influence the development cycle, reduce overall integration efforts and provide the necessary development tools to support the SENET-DAX development effort.

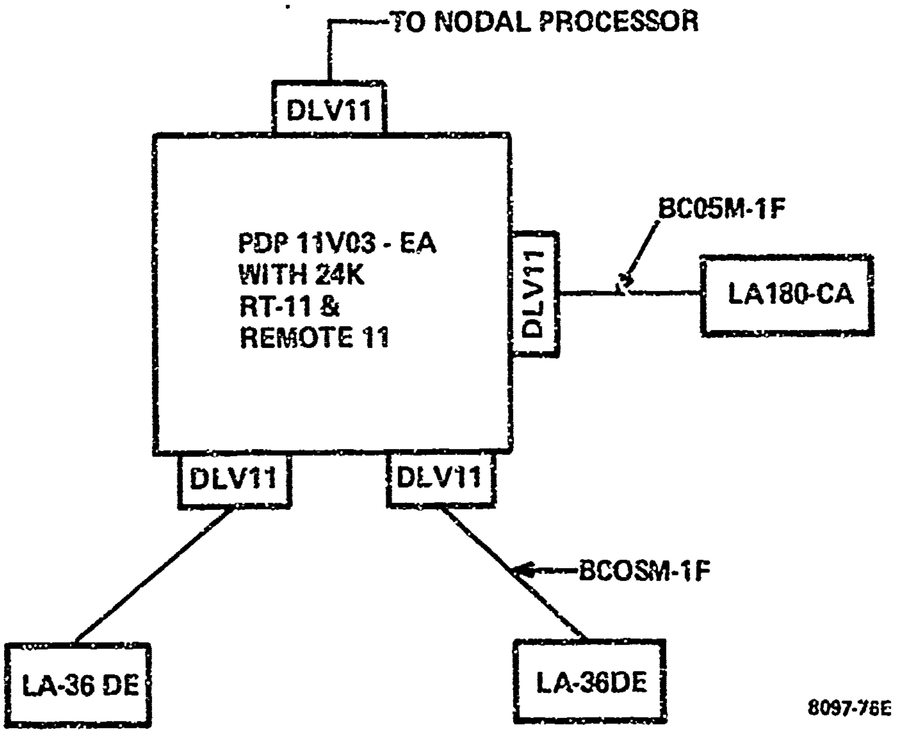


Figure 4-2. Off-line Software Development System

SECTION 5

AN EXPERIMENTAL INTEGRATED VOICE/DATA SWITCHING NETWORK

5.1 GENERAL

The recommended conceptual model will be used to demonstrate the practicality of the SENET-DAX concept in a realistic network environment. Figure 5-1 illustrates three network configurations. Figure 5-1A shows a two-node network consisting of two SENET-DAX switches connected by a 230.4 kBPS link, with a limited number of directly connected digital subscribers at each node. The two-node network could support the demonstration of most of the basic concepts and services listed in Table 1-1. In some cases, however, the demonstration would be limited. For example, the various levels of packet switching protocol would be difficult to demonstrate as would a realistic case of alternate routing of either Class I or Class II data.

Figure 5-1B illustrates a three-node fully connected network with end and tandem node capability and a limited number of direct subscribers. As in the two-node network, the connecting links are 230.4 kBPS full-duplex. This three-node network could support the demonstration of all the basic concepts and services listed in Table 1-1, in addition to alternate routing performed by an originating switch and spill backward routing at a tandem switch. With three nodes some capability is given up, although all of the basic services can still be demonstrated.

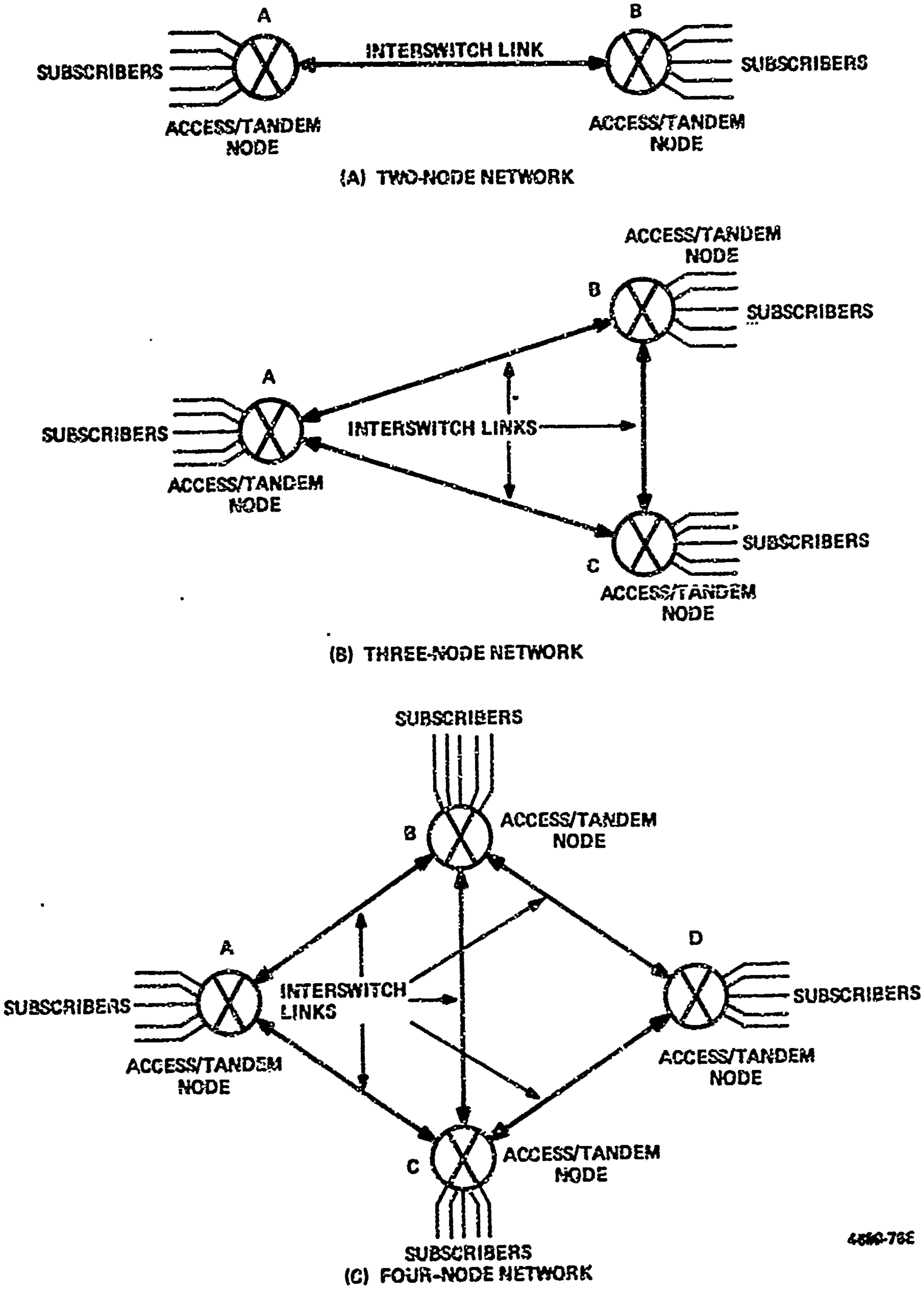


Figure 5-1. Experimental Network Configurations

Figure 5-1C illustrates a four-node network having two fully connected nodes and two nodes each connected to the remaining nodes but not to each other. All nodes have direct subscribers and handle trunk-to-trunk traffic. This network was described more fully in paragraph 1.3. The four-node experimental network is recommended as the most suitable for exploring the concept for a working model based on the SENET-DAX technique. A four-node network allows demonstration of all the fundamental services in Table 1-1, as well as allowing greater possibilities for alternate routing and recovery from blocking than in the three-node network. The addition of more nodes (i.e., a five or six-node network) would not provide significant additional capability, except minimally with respect to alternate routing, and would increase total cost of the network. It appears that the four-node network is the most effective in providing the services necessary to demonstrate workability of the model.

Other advantages of the four-node configuration are that it provides a multiplicity of alternate routes, and will respond more efficiently to local overload at selected nodes. Assume that a call is made from a subscriber at switch A to a subscriber at switch C. In the three-node network there is a primary path along one link (A-C) and an alternate path which transits two links (A-B-C). With the four-node network there would be a choice of two 2-link paths (A-B-D and A-B-C) and two 3-link paths (A-B-C-D and A-C-B-D). One could also envision an experiment using the traffic generator which could simulate a situation in which displacement of subscribers results in a temporary distribution unbalance and concentration of subscribers at particular nodes. The ability of the network to adjust to this situation could be evaluated. Also, the enhancement of survivability when a node is lost could be tested by the use of diagonal links.

Cost estimates for the two, three, and four-node configurations are provided in Section 7. If network costs for the four-node design preclude its use at this stage, then the three-node alternative could provide most of the features that are critical to the evaluation. The two-node network could also demonstrate that the basic concept will work; later on, two more nodes could be added and additional features such as alternate routing could be tested. The advantage of this approach is that once the software is developed for the two-node configuration , most, if not all, of the development work has been completed (this does not include software developed for routing). The transition from a two-node to four-node network should then take a relatively short time.

5.2 CAPABILITIES

The four-node network test configuration provides a means of examining the integrated system concept for an access area application. All of the basic SENET-DAX conceptual ideas in paragraph 1.3 and the services in Table 1-1 can be demonstrated. The establishment and maintenance of various types of call operations could demonstrate system feasibility and flexibility. Possibilities include:

a. Class I voice calls between switching centers, e.g., at rates of 16 kBPS, 2400 kBPS or others

b. Class II interactive terminal/terminal and computer/terminal calls between switches. Possibilities for computer communication include GTE time sharing system, or the ARPANET. A rudimentary computer/terminal call can be simulated by substituting the nodal processor for a teletype at one of the switch locations

c. Simultaneous ClassI/Class II operation between switches. Typical calls might include a Class I voice call at 16 kBPS, a Class I voice call at 2400 kBPS and a Class II interactive computer/terminal call.

The model will also provide an investigatory tool for examining the feasibility of other more advanced functional network concepts in the future. These might include:

a. Preemption of a Class I call by a Class II packet either by the technique of dropping the call or causing it to be "interrupted" for a frame in order to transmit the packet

b. Preemption of Class II data, e.g., a packet, by a Class I call, with corresponding delay of the packet at the node where preempted, or re-routing of the packet elsewhere through the network

c. Mixed service for voice/data terminals. This could consider voice and data multiplexed and switched together as a fixed increment in the Class I region as well as a hybrid procedure which identifies the particular service required (voice or data) and connects the corresponding terminal to the proper interface circuitry

d. Dynamic alternate routing with precedence.

5.3 DEMONSTRATION DISPLAYS

Several displays could be used to illustrate the basic switching concept and implementation techniques. Among these are computer printouts and oscilloscope waveform displays. Printouts of the channel map at a switching center would illustrate the dynamic allocation of various channel widths to subscribers having different terminal rates, and the drop, insert, and recompacting of channels during call setup and release. Call setup, with its inherent forward reservations of channel links during the search phase and backward allocation subsequent to the called party answering, would also be illustrated effectively by printing out successive "snapshots" of network link allocation. Alternate routing and quasi-associative routing can be likewise displayed. Oscilloscope waveform displays at the trunks and subscriber terminals with appropriate signal generator inputs applied to the end terminals can be used to dramatize the dynamic allocation of a master frame and the interaction between channels of different widths. Specific patterns generated at a subscriber terminal may be viewed on the trunk, as the Class I/Class II boundary dynamically shifts in reaction to the allocation of bytes appended to the Class I portion of the frame. The positional allocation of subsequent calls and the recompacting of current call allocations as previous calling parties go on-hook, can also be shown.

SECTION 6

PERFORMANCE EVALUATION

6.1 GENERAL

In this section the merits of the distributed processing model are examined as a function of switching capabilities. This included the evaluation of cross-network and cross-office delays, an assessment of system capabilities with regard to traffic handling, throughput, and the amount of control overhead requiring transmission capacity, and an analysis of blocking and delays. Where applicable, the constraints and limitations on switching capability, as a result of compromises necessary in order to develop the experimental system, will be assessed.

6.2 PARAMETRIC ANALYSIS OF TRANSMISSION EFFICIENCY AND THROUGHPUT

6.2.1 Transmission Efficiency and Throughput Defined

Transmission efficiency E_{ff} is defined as the ratio of correct information ITR* transmitted over the trunk to the trunk transmission rate R. The trunk transmission rate R can be further subdivided into the sum of the correct information exchange rate ITR plus the transmission rate of the overhead R_O:

$$E_{ff} = \frac{ITR}{R} = \frac{ITR}{ITR + Ro} \qquad (6\text{-}1)$$

* ITR is also known as the throughput of the system.

Although Class I and Class II data are transmitted simultaneously over the same trunk, ITR and R_O are different for each case. The overall transmission efficiency is defined as the weighted average of the Class I and II efficiencies with the weighting factors being the percentage of each class of traffic in the total trunk traffic. Thus,

$$E_{ff_T} = \frac{E_I \ ITR_I + E_{II} \ ITR_{II}}{(E_I + E_{II}) \ R} \tag{6-2}$$

where E_I and E_{II} are the Class I and Class II traffic, respectively and ITR_I and ITR_{II} are the respective Class I and Class II throughput. Note that if $E_I = 0$, no Class I traffic is transmitted and the efficiency

$$E_{ff_T} = \frac{ITR_{II}}{R}$$

becomes that for Class II only. The overall transmission overhead rate is thus given by

$$Ro_T = \frac{E_I \ Ro_I + E_{II} \ Ro_{II}}{E_I + E_{II}} \tag{6-3}$$

and the overall throughput by

$$ITR_T = \frac{E_I \ ITR_I + E_{II} \ ITR_{II}}{E_I + E_{II}} \tag{6-4}$$

where

$ITR_I = Eff_I$ RI is the rate of correct information transmitted in the Class I region,

R_I is the effective transmission rate for Class I data,

$ITR_{II} = Eff_{II} R_{II}$ is the rate of correct information transmitted in the Class II region, and

R_{II} is the effective transmission rate for Class II data.

In the rest of this section we will extend the analysis undertaken in Phase I and re-examine the characteristics of transmission overhead and throughput for Class I and II traffic in the experimental network described in this report. Then we will investigate the Class II efficiency and throughput for the lower carrier rate of 230.4 kBPS, parametrically, as a function of packet size, bit error rate and error control protocol.

6.2.2 Class I and II Traffic Overhead

The portions of the dynamically allocatable SENET-DAX frame which are considered to be overhead in the experimental system will now be defined. In the Class I region CCIS messages used to establish connections and dynamically control the position of any channel within the region are considered to be Class I overhead. This includes the messages used in the establishment and termination of calls, coordination and control of the frame, messages to compact the allocation of bits in the frame after a call is terminated, and other control messages which utilize the CCIS subregion, such as call synchronization and resynchronization messages, acknowledgments, routing control and maintenance messages. Other types of overhead information considered to be Class I overhead are the 16-bit start-of-frame markers for frame synchronization, the 48-bit frame marker for resynchronization and the overhead due to bit stuffing because of the 8-bit slot size. All other data transmitted in the Class I region will be considered as effective throughput regardless of content or transmission errors. This could include data consisting of voice, noise, or idle periods, or facsimile with forward error control (FEC).

During the Phase I study it was found that CCIS messages, the SOF marker signifying the start-of-frame, and bit stuffing because of the 8-bit slot size, were a small fraction of the total overhead (on the order of 1 percent). This was examined for the case of a T1 carrier rate (1.544 MBPS) and a 10 msec frame. A switch capable of serving a maximum of 300 voice subscribers at a 0.1 percent grade-of-service and 372 data subscribers was derived from these system parameters. Here we consider an experimental model which utilizes a lower trunk transmission rate (230.4 kBPS); however, it only has the capability of serving a maximum of 20 voice subscribers and 360 data subscribers at the same grade-of-service requirements. Therefore, we expect the percentage of Class I overhead to remain a small proportion of the total. For this reason further examination of Class I overhead was curtailed and attention directed to the Class II region where the effect of using a lower transmission rate was evaluated for Class II throughput and efficiency.

Class II overhead consists of several major categories. One category concerns the non-information bits of a packet, such as those bits in the flag, address, control or frame check sequence fields, that are transmitted over the trunk along with bits of the information field. These provide a means of exchanging data between nodes in a network and include the functions of message, packet and network error control, sequencing, supervisory control, addressing, and additional link control functions. The retransmission of packets which are delivered in error is also catagorized as overhead.

A third category concerns the idle time spent by the trunk while waiting for an acknowledgment (ACK) that the packet just transmitted has been received correctly, as is the case of block-by-block ARQ. For continuous ARQ-Type I, where all packets transmitted in the intervening time between the transmittal of the erroneous packet and receipt of the negative acknowledgment (NACK) are retransmitted, this overhead consists of the time necessary for retransmitting the correct packets.

Class II transmission efficiency and throughput are a function of the error environment, packet size and structure, ARQ error protocol, and the transmission rate of the system. During Phase I, equations for transmission efficiency were derived and analyzed to determine an optimum packet size that maximized the efficiency for a given error environment. The results demonstrated that transmission efficiency increases with improved error environment and exhibits a resonance phenomenon for variations in packet size, with the optimal packet size for greatest efficiency increasing at roughly one-third the rate of improvement in the error environment. In this report we examine these results for the lower transmission rate of 230.4 kBPS and extend the analysis to include the Class II throughput.

6.2.3 Optimization of Class II Transmission Efficiency (E_{ff}) and Throughput (ITR)*

The parameters which are used in the derivation of the transmission equations are listed below. These include:

E_B = Probability of a bit being incorrect

DC = Duty cycle of noise burst, i.e., percentage of time during which a burst condition exists

P = Packet size, composed of I information bits and C control bits (includes bits from the flag, address and control fields, frame check sequence and/or parity bits, sequence numbers, stuffing bits for filling out words or blocks, and packet length or I.D. features)

R_{II} = Class II trunk transmission rate

T = Time spent in transmitting a packet

W = Time spent waiting for a transmitted packet to be acknowledged.

*For convenience of notation, in the rest of the report, E_{ff} and ITR will be taken to be the Class II transmission efficiency and throughput, respectively.

Error control is an ARQ form of protocol whereby the receiving link processor monitors the incoming signal for transmission errors and notifies the transmitting processor when it is necessary to repeat the packet. The transmitter is notified via a control message over the return link. Three types of ARQ error control were considered: block-by-block ARQ and two types of continuous ARQ. In block-by-block ARQ, after transferring a block, the trunk remains idle while waiting for the acknowledgment in order to transmit the next block of information. In continuous types of ARQ systems, there is no waiting time for acknowledgment after a packet is sent. As soon as a packet is transmitted the next one is sent. When a NACK is received for a particular packet, either all packets transmitted in the intervening time between the original transmission of the erroneous packet and the receipt of the NACK are retransmitted (ARQ-Type I) or only the erroneous packet in question need be transmitted (ARQ-Type II). The latter case, where only the packet that was detected in error is repeated, produces more efficient operation than either of the other two ARQ schemes.

Class II information is transmitted across the trunk in packets of P bits each. The ratio of non-overhead bits to the total packet size is given by $\frac{I}{P}$ where P = I+C. Sometimes a bit is in error, causing the packet to be in error. This causes the efficiency to be modified according to

$$\frac{I}{[P/(1-Ep)]},$$

where

$$E_P = 1-(1-E_B)^P.$$

E_B is the probability of a bit being incorrect, $(1-E_B)^P$ is the probability of a packet being correct, and E_P is the probability of a packet being in error in a random error environment. For small values of E_P

$$1 - E_P = (1-E_B)^P \simeq \varepsilon^{-PE_B}.$$

If the channel is subjected to error bursts, the effect during the time of the burst is to render the channel useless. The channel is less efficient in proportion to the time the burst condition exists, i.e., $\frac{I}{[P/(1-E_P)(1-DC)]}$.

The effect of a noise burst will be to delay the transfer of information while waiting for acknowledgments. In block-by-block ARQ the transmitter is idle from the time it sends the last bit of a packet until it receives and interprets the return ACK or NACK message. The efficiency becomes

$$\frac{I}{[P/(1-E_P)(1-DC)]} \cdot \frac{T}{T+W} ,$$

where

$$T = \frac{P/(1-DC)}{R}$$

is the time it takes to transmit one packet during the time that the channel is usable and W is the waiting time per packet. Taking all this into account, the transmission efficiency for block-by-block ARQ is given by the expression

$$E_{ff} = \frac{IE^{-PE_B}(1 - DC)}{I + C + WR_{II}(1 - DC)} \qquad (6\text{-}5)$$

The throughput is this efficiency multiplied by the channel Class II transmission rate, i.e.,

$$ITR = E_{ff}\,R_{II} = \frac{R_{II}\,IE^{-PE_B}(1 - DC)}{I + C + WR_{II}(1 - DC)} \qquad (6\text{-}6)$$

By differentiating equation (6-5) with respect to the packet size and setting the result equal to zero, $\frac{dE_{ff}}{dP} = 0$

the optimum packet size can be determined and is given by the solution of the equation

$$I^2 + [C + WR_{II}(1 - DC)]\,I - \frac{[C + WR_{II}(1 - DC)]}{E_B} = 0 \qquad (6\text{-}7)$$

Solving for I we find

$$I_{opt} = \frac{[C + WR_{II}(1 - DC)]}{2E_B}\left\{\sqrt{1 + \frac{4}{[C + WR_{II}(1 - DC)]E_B}} - 1\right\} \qquad (6\text{-}8)$$

and

$$P_{opt} = I_{opt} + C \qquad (6\text{-}9)$$

In the continuous ARQ-Type I mode where all data following the packet in error, is retransmitted, a waiting period is suffered only when an error occurs. The efficiency thus becomes

$$\frac{I}{[P/(1-Ep)(1-DC]} \cdot \frac{T}{T+W(1-\varepsilon^{-PE_B})}$$

where

$$W(1-E^{-PE_B})$$

represents the non-productive time during which the channel is idle because the packet is in error and data must be resent. In this case the transmission efficiency is given by

$$E_{ff} = \frac{I\varepsilon^{-PE_B}(1 - DC)}{I + C + WR_{II}(1-DC)(1 - \varepsilon^{-PE_B})} \qquad (6\text{-}10)$$

and the throughput by

$$ITR = \frac{R_{II}\, I\varepsilon^{-PE_B}(1 - DC)}{I + C + WR_{II}(1 - DC)(1 - \varepsilon^{-PE_B})} \qquad (6\text{-}11)$$

These parameters are maximized by differentiating with respect to I and equating to zero. I_{opt} is then the solution of the expression

$$E_B I^2 + [C + WR_{II}(1 - DC)]E_B I - [C + WR_{II}(1 - DC)] + WR_{II}(1 - DC)\varepsilon^{-PE_B} = 0 \qquad (6\text{-}12)$$

which may be solved by the method of successive approximations [2]. Results of this are shown in paragraph 6.2.4.

For the continuous ARQ-Type II mode, where only the packet in error is retransmitted, the efficiency is modified according to the expression

$$\frac{I}{[P/(1-Ep)(1-DC)]} \cdot \frac{T}{T+\frac{P}{R}(1-\varepsilon^{-PE_B})}$$

where

$$\frac{P}{R}(1-\varepsilon^{-PE_B})$$

represents the non-productive time spent in transmitting a packet which is in error and therefore must be considered to be pure overhead. The transmission efficiency thus becomes

$$E_{ff} = \frac{I\varepsilon^{-PE_B}(1 - DC)}{I + C + P(1 - DC)(1 - \varepsilon^{-PE_B})} \qquad (6\text{-}13)$$

and the throughput

$$ITR = \frac{R_{II}\ I\varepsilon^{-PE_B}\ (1 - DC)}{I + C + P\ (1 - DC)\ (1 - \varepsilon^{-PE_B})} \qquad (6\text{-}14)$$

The method of solution is the same as before and leads to the following expression

$$(2\text{-}DC)\ E_B I^2 + (2\text{-}DC)\ CI - (2\text{-}DC)\ C + (1\text{-}DC)\ C\varepsilon^{-PE_B} = 0 \qquad (6\text{-}15)$$

which also may be solved by successive approximation [2], the results of which will be shown in paragraph 6.2.4

6.2.4 Variation of E_{ff} and ITR with Packet Size and Bit Error Rate

Using typical SENET-DAX baseline parameters, a study of the variation of transmission efficiency, throughput, optimized packet size, and other performance parameters was conducted for varying values of bit error rate for block-by-block ARQ and the two types of continuous ARQ. The maximum Class II transmission rate was 44.8 kBPS, which corresponded to approximately 20 percent of the trunk capacity of the 230.4 kBPS carrier used in the experimental system. The results are shown in Figures 6-1 through 6-6 and in Table 6-1. Continuous ARQ provides a greater efficiency (throughput) than block-by-block ARQ, but requires greater data capacity. Continuous ARQ-Type II, where only the erroneous packet is repeated, produces the greatest improvement in efficiency (throughput) for the error environments investigated. For a constant packet length, efficiency (throughput) improves as Bit Error Rate (BER) improves (see Figures 6-1, 6-2, and 6-3). At low BER's (10-7), the larger the packet size, the better the transmission efficiency, while at high BER's (10-3)

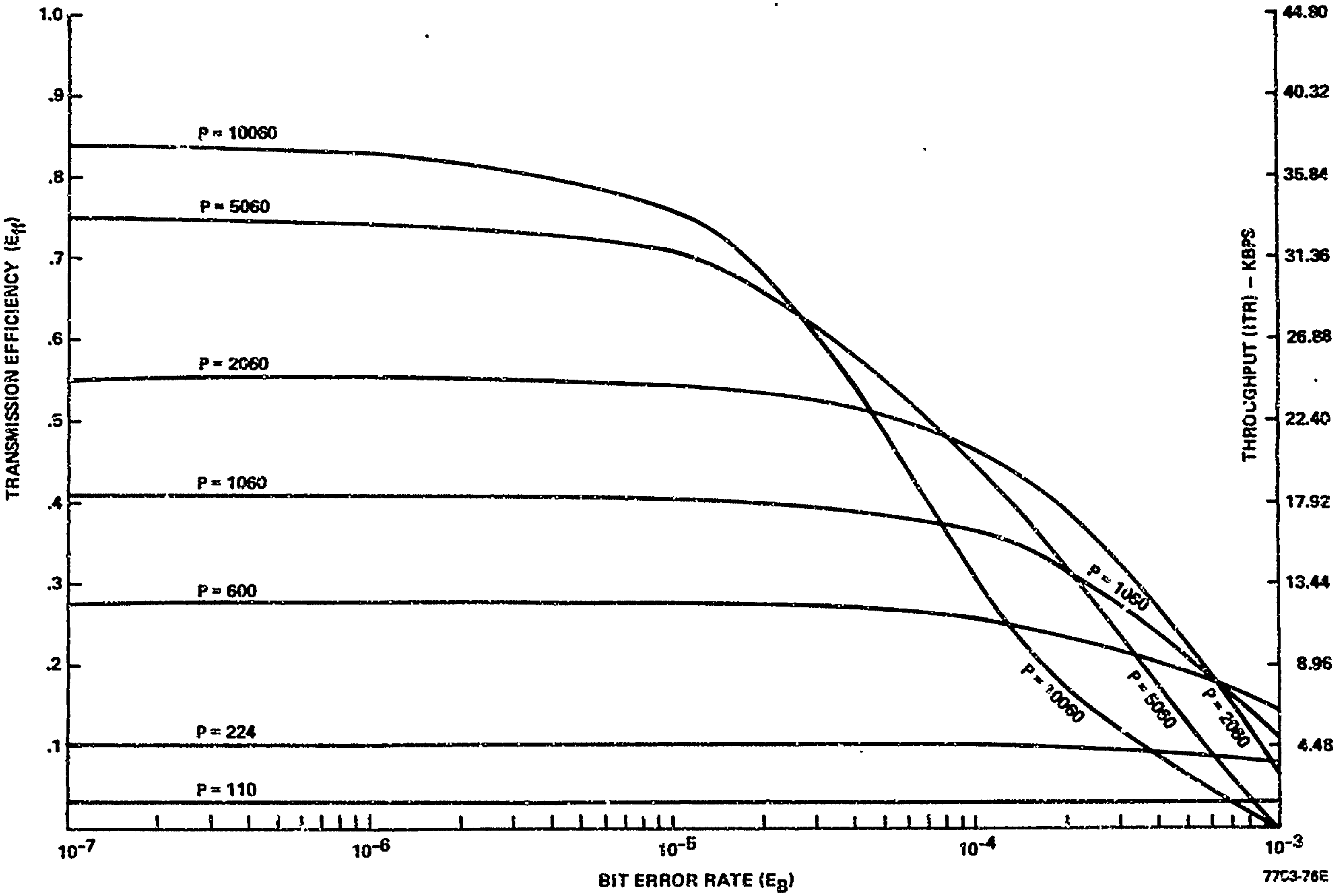


Figure 6-1. Transmission Efficiency vs. Bit Error Rate Block-by-Block ARQ

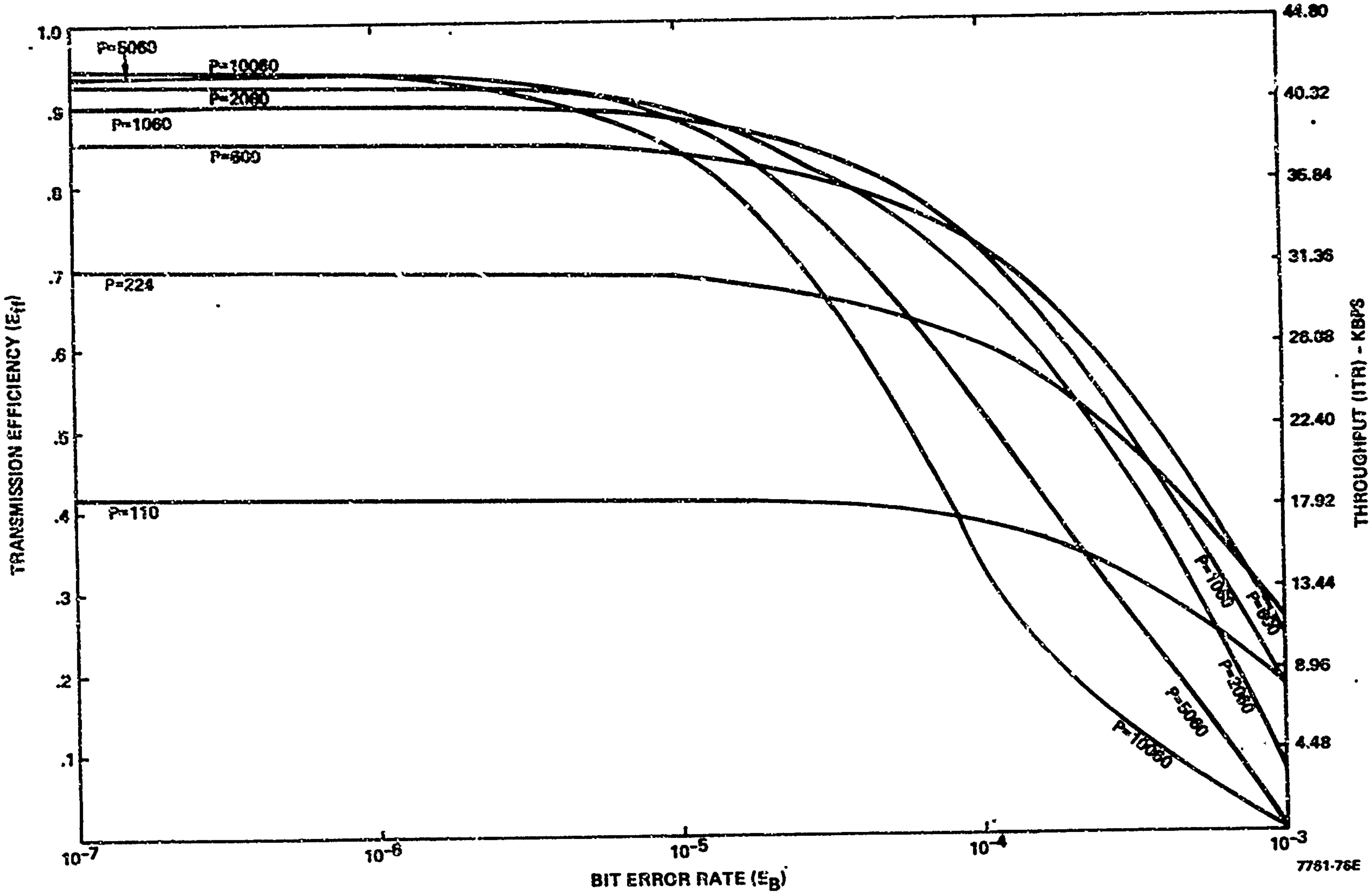


Figure 6-2. Transmission Efficiency vs. Bit Error Rate Continuous ARQ-Type I

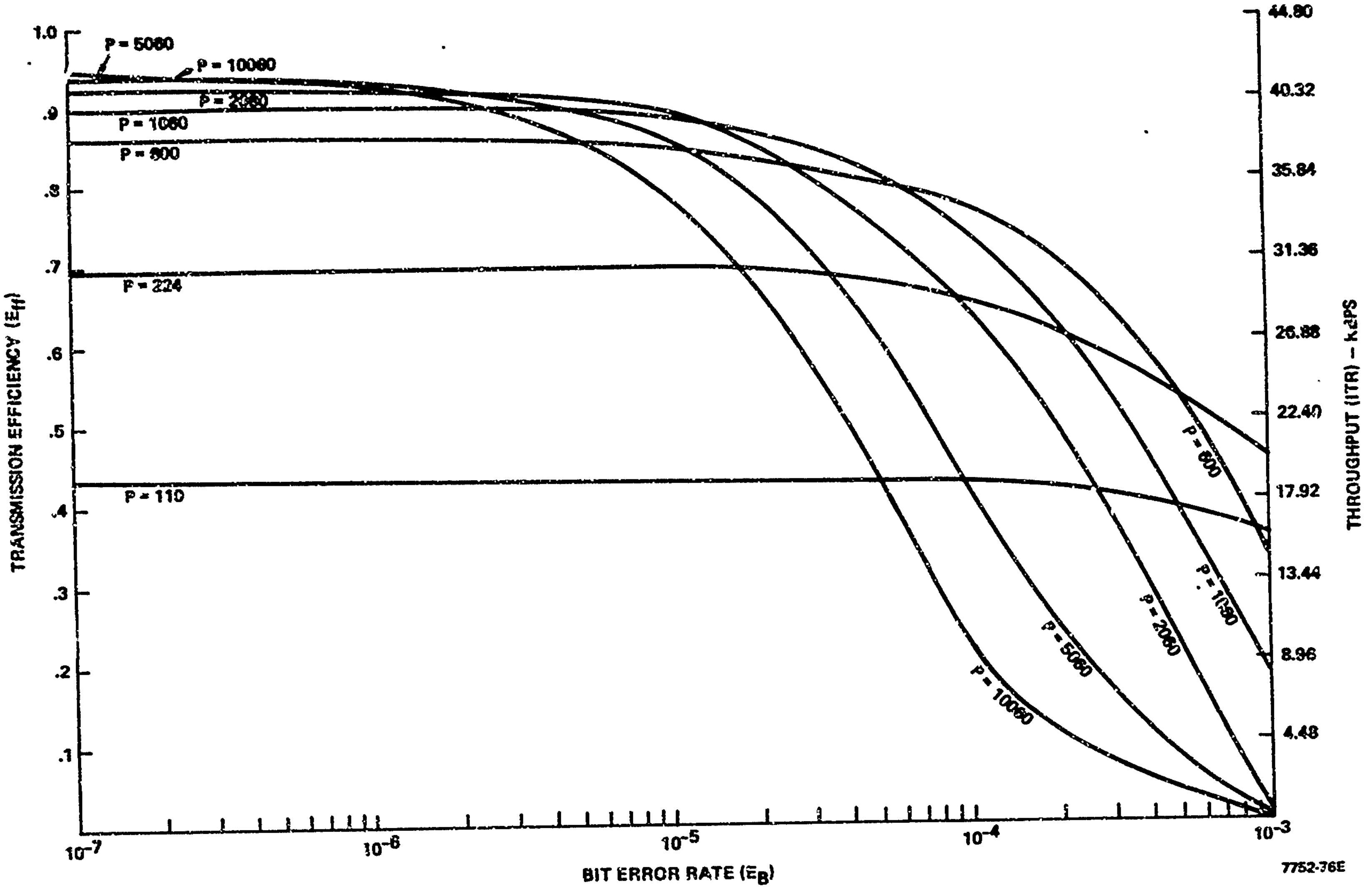


Figure 6-3. Transmission Efficiency vs. Bit Error Rate
Continuous ARQ-Type II

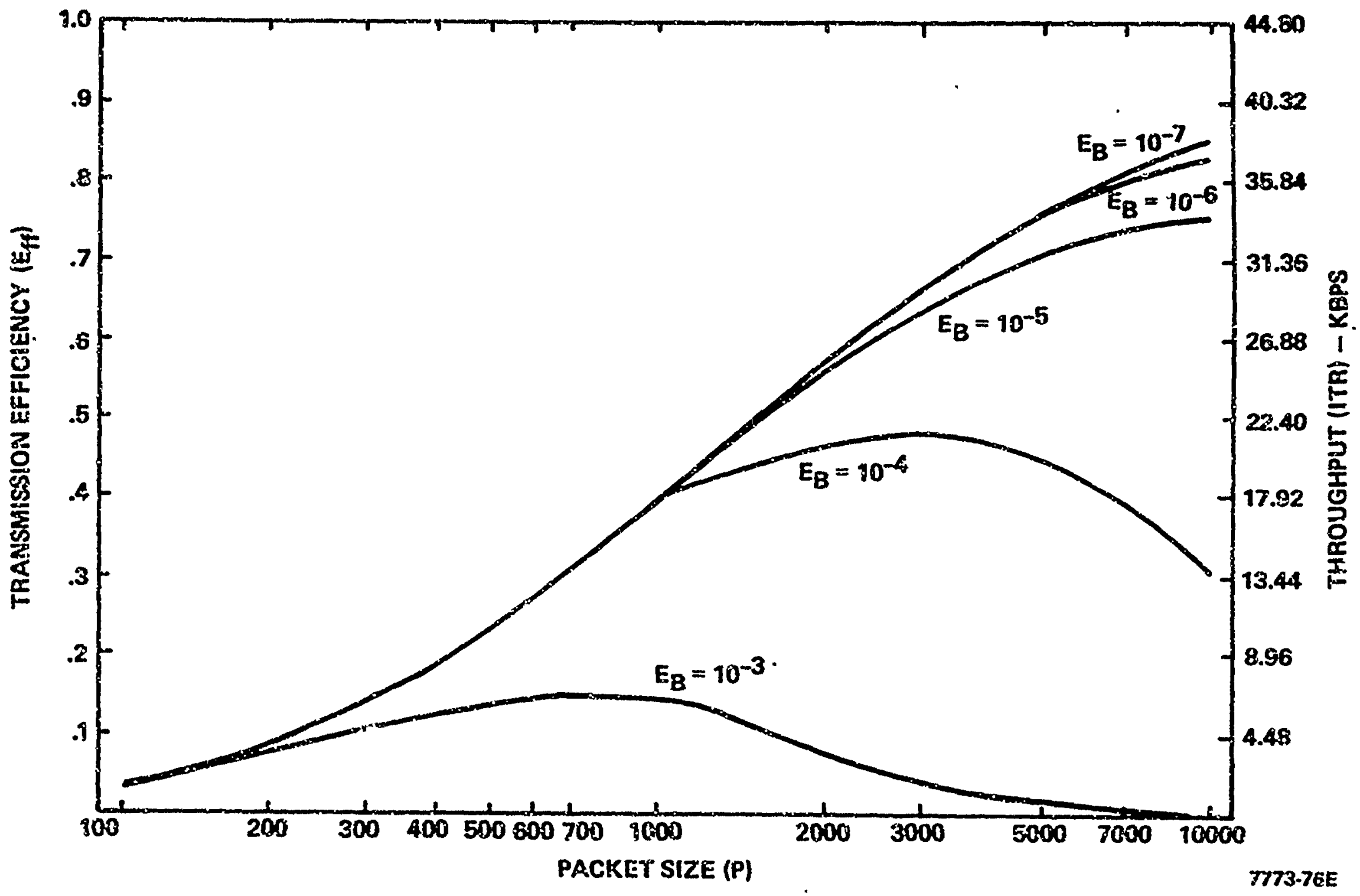


Figure 6-4. Transmission Efficiency vs. Packet Size Block-by-Block ARQ

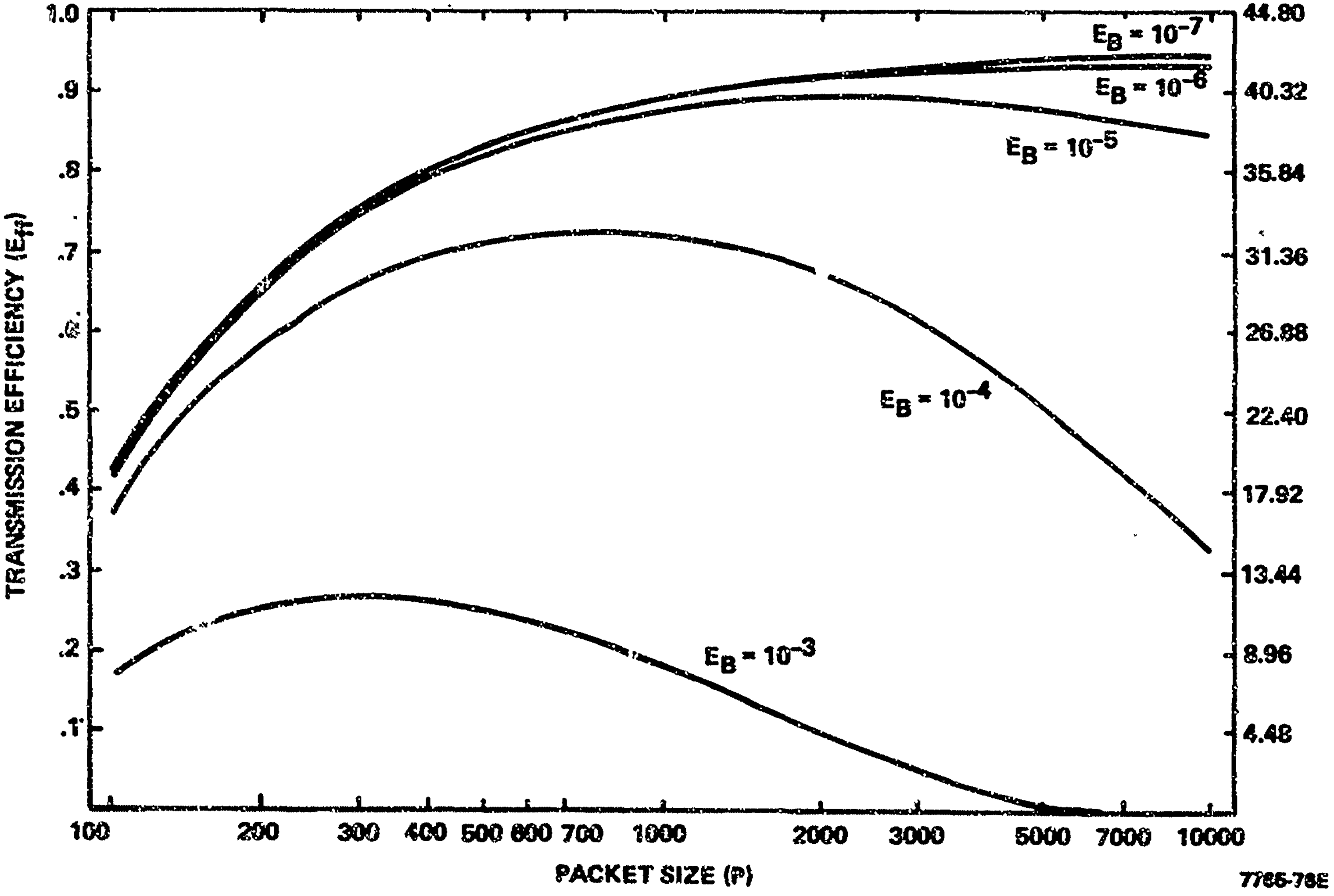


Figure 6-5. Transmission Efficiency vs. Packet Size
Continuous ARQ-Type I

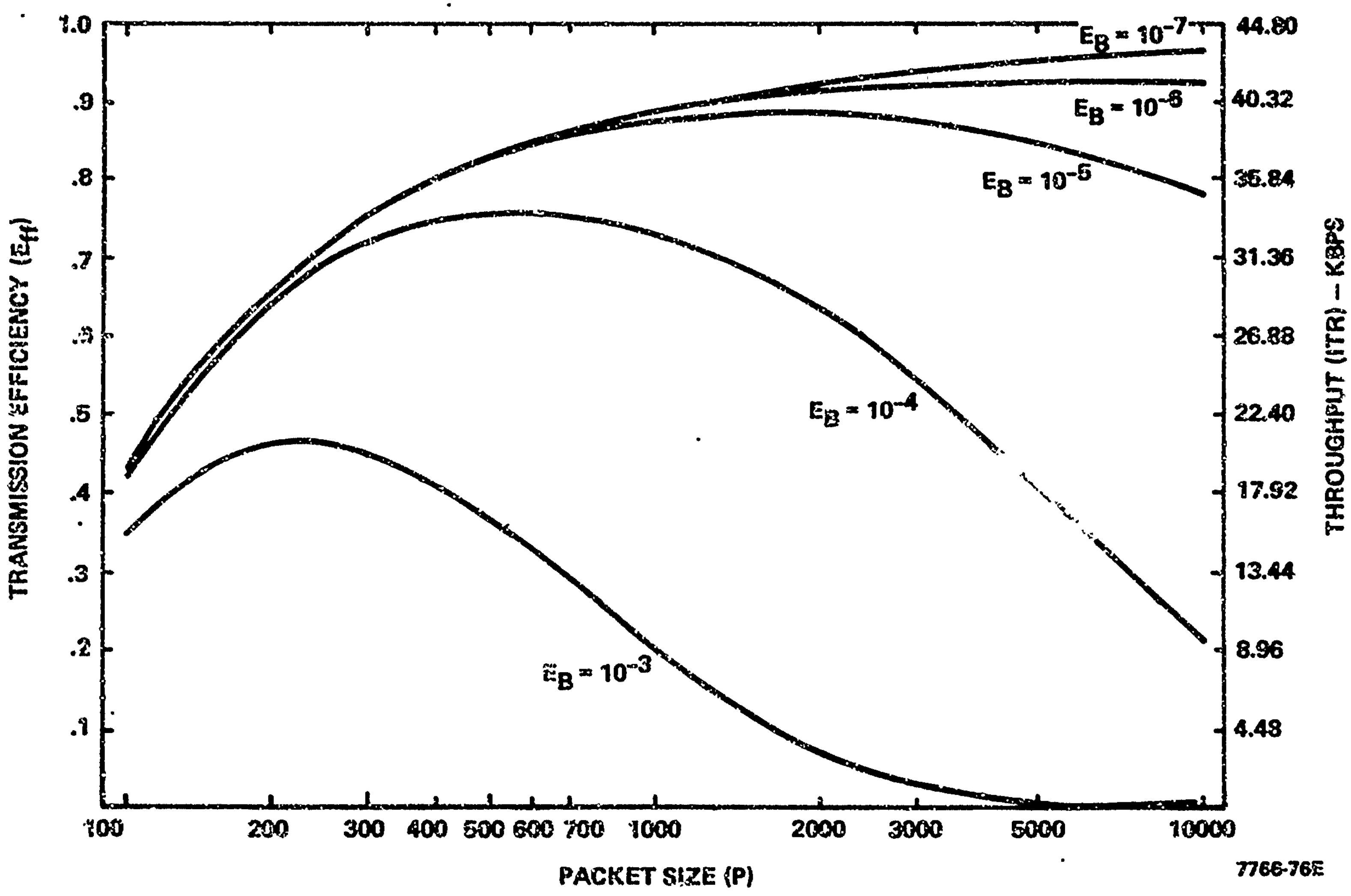


Figure 6-6. Transmission Efficiency vs. Packet Size Continuous ARQ-Type II

TABLE 6-1. OPTIMIZATION OF TRANSMISSION EFFICIENCY AND THROUGHPUT FOR DIFFERENT ARQ PROTOCOLS

ARQ PROTOCOL	BIT ERROR RATE	OPTIMUM PACKET SIZE (BITS)	OPTIMUM TRANSMISSION EFFICIENCY	OPTIMUM THROUGHPUT (KBPS)	PACKET ERROR RATE
BLOCK-BY-BLOCK	10^{-3}	726	.1617	7.24	.5111
	10^{-4}	3,109	.4840	21.68	.2672
	10^{-5}	10,977	.7583	33.97	.1040
	10^{-6}	35,753	.8836	39.59	.0353
	10^{-7}	115,025	.9284	41.59	.0114
CONTINUOUS TYPE I	10^{-3}	313	.2744	12.29	.2614
	10^{-4}	827	.7226	32.37	.0794
	10^{-5}	2,485	.8930	40.01	.0245
	10^{-6}	7,785	.9342	41.85	.0078
	10^{-7}	24,360	.9452	42.34	.0024
CONTINUOUS TYPE II	10^{-3}	224	.467	20.92	.2007
	10^{-4}	600	.763	34.18	.0582
	10^{-5}	1,798	.8969	39.73	.0178
	10^{-6}	5,586	.9296	41.65	.0056
	10^{-7}	17,536	.9435	42.27	.0018

7778-76E

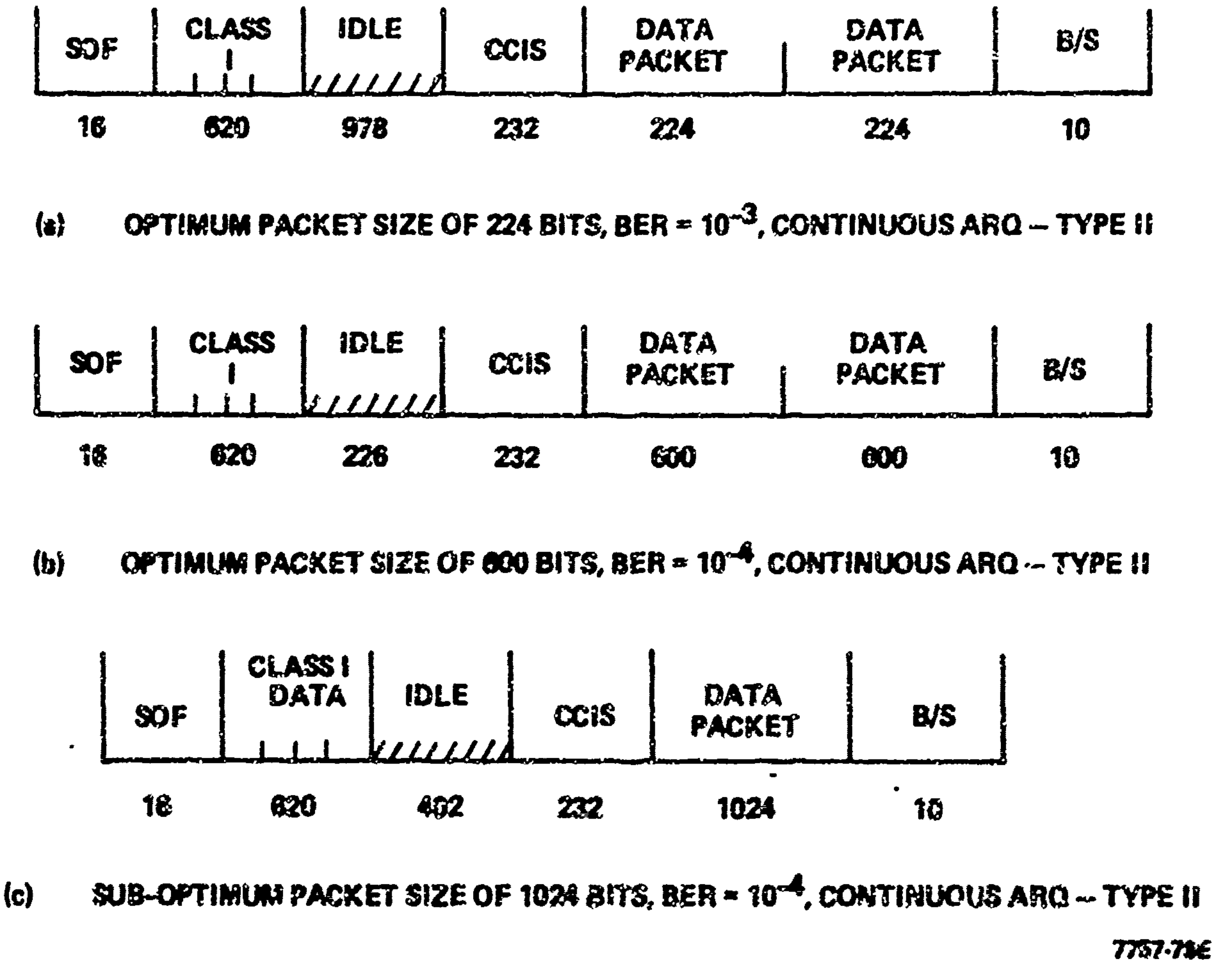
just the opposite occurs; the larger packets have poorer efficiency (throughput) ratings. The reasons for this are obvious. As more bits are transmitted per packet, the transfer becomes more efficient. At high BER's, a considerable number of errors occur, causing retransmission of erroneous packets. All packets with errors in them are considered to be pure overhead. Therefore, the more bits transmitted per packet, the less likely it is of reaching its destination in a bad error environment, hence the more inefficient the transfer. Likewise, the smaller the packet size, the more likelihood there is of it being received without error.

For a constant error environment, as the packet length is increased, an optimal efficiency (throughput) is reached (see Figure 6-4, 6-5, and 6-6). Increasing the packet size beyond this value results in a decrease in efficiency (throughput) for the reasons noted above. Thus a resonance effect is observed. This optimum occurs at a different packet size for different BER's. As the error environment becomes more error-free, the maximum transmission efficiency (throughput) that the system can attain increases and the larger the optimum packet size becomes. At high bit error rates it is more efficient to transfer small-sized packets; however, the maximum efficiency (throughput) obtainable is somewhat limited. At low BER's it is more efficient to transmit larger packets. As the BER decreases, the maximum transmission efficiency (throughput) obtainable approaches unity in the limit.

The same information can be displayed in tabular form (see Table 6-1). A packet is composed of information bits (I) and control bits (C). Equations (6-8), (6-12), and (6-15) are solved for I_{opt}, the optimum packet size, and substituted into equations (6-5), (6-6), (6-10), (6-11), (6-13), and (6-14) to obtain Eff_{opt}, the optimum transmission efficiency, and ITR_{opt}, the optimum throughput, for a given BER. As the error environment improves, the optimum packet size becomes larger and the transmission efficiency approaches unity. Likewise, the Class II throughput approaches its maximum of 44.8 kBPS. For continuous ARQ-Type II, at a BER of 10^{-3}, the optimum packet size is 224 bits, the transmission efficiency is 46.7 percent and the throughput is 20.92 kBPS (see Table 6-1). This is 3 times better than block-by-block ARQ and 1.7 times better than continuous ARQ-Type I. The probability of a packet being received in error is 20.07 percent.

6.2.5 Data Utilization as a Function of Voice Traffic

Figure 6-7 shows the allocation of bits in a frame for several packet sizes with continuous ARQ protocol, Type II, i.e., repeating all data from a NACK. In paragraph 6-3 it will be shown that for a packet size of 224 bits (optimum for a 10^{-3} BER), a busy hour data load of 200 packets per second (or 2 packets/frame) can be accommodated along with an offered voice traffic load of 4 erlangs, with negligible queuing delay (less than .1 msec per packet) due to traffic congestion. With this arrangement (Figure 6-7a) a grade-of-service of .1 percent can be guaranteed for the 4 erlangs of voice traffic, and one CCIS message (maximum number of bits = 232) can be carried simultaneously with the voice and data. The efficiency



(a) OPTIMUM PACKET SIZE OF 224 BITS, BER = 10^{-3}, CONTINUOUS ARQ – TYPE II

(b) OPTIMUM PACKET SIZE OF 600 BITS, BER = 10^{-4}, CONTINUOUS ARQ – TYPE II

(c) SUB-OPTIMUM PACKET SIZE OF 1024 BITS, BER = 10^{-4}, CONTINUOUS ARQ – TYPE II

7757-79E

Figure 6-7. Allocation of Traffic in a Frame Interval for Various Packet Lengths

attained with an optimum packet size of 224 bits is .467 while that attained with a packet size of 600 bits is .763. It is interesting to note from Figure 6-6 that the efficiency for the sub-optimum packet size of 1024 bits (which is the maximum packet length for the ARPA Network) is equal to .735. Therefore, even though an optimum packet size can be specified, efficiencies do not vary significantly over a wide range of packet sizes since the curves are relatively flat for most of the error rates in Figure 6-6.

Table 6-2 shows the variation of the link data capacity as a function of the voice traffic laod for the optimum packet size of 224 bits (Figure 6-7a). For the busy hour data load of 2 packets per frame and an offered traffic load of 4 erlangs, the percentage of the frame available for packet data transmission is 61.89 percent. This falls to 28.9 percent when overhead due to noise is taken into account. The data utilization factor, which represents how much of the realizable throughput is used for transmission of the busy hour data load, is 67.28 percent. This corresponds to a total trunk utilization of 79.77 percent. It can be seen from Table 6-1 that a maximum of 7 erlangs of voice can be tolerated and still retain the capacity to carry the busy hour data load. The <u>peak</u> trunk utilization factor then is 99.84 percent of the link data capacity during the busy hour, or 99.74 percent of the total trunk capacity. Decreasing the peak data load, e.g., to a rate of 100 packets/sec, would allow more voice traffic to be carried.

TABLE 6-2. DATA UTILIZATION AS A FUNCTION OF OFFERED VOICE TRAFFIC

BUSY HOUR DATA LOAD (2 PACKETS)	OFFERED VOICE TRAFFIC (ERLANGS)	PERCENT OF FRAME AVAILABLE FOR DATA (ERROR FREE)	PERCENT OF FRAME AVAILABLE FOR DATA (TAKING INTO ACCOUNT) CHANNEL ERRORS)	DATA UTILIZATION FACTOR	TOTAL TRUNK UTILIZATION
19.44%	0	88.80%	41.48%	46.87%	52.86%
	1	82.07	38.33	50.72	59.59
	2	75.35	35.18	55.26	66.32
	3	68.62	32.05	60.66	73.05
	4	61.89	28.90	67.28	79.77
	5	55.16	25.76	75.47	86.50
	6	48.44	22.62	85.94	93.23
	7	41.71	19.47	99.84	99.74
	8	34.98	16.33	OVERLOAD	OVERLOAD

7750-76E

This type of chart illustrates the upper bound on voice and data packets carried simultaneously and provides a means for analyzing how one type of traffic can be increased at the expense of the other. For a constant packet size, as the bit error rate decreases, the more efficient the transfer of information, and the lower the peak trunk data utilization (the trunk is used less for retransmission of incorrect packets). However, as the packet size is increased, the upper limit on voice traffic carried is realized sooner. For example, at a bit error rate of 10^{-4}, an optimum packet size of 600 bits is specified. Taking into account the effects of noise, the busy hour data load of 2 packets per frame and 4 erlangs of voice would result in an overload condition. However, we could transmit one packet per frame and 4 erlangs of voice simultaneously with a data utilization factor of 55.15 percent and a total trunk utilization of 72.24 percent.

6.2.5 Throughput and Efficiency as a Function of Transmission Rate

Several interesting points come to light when we compare the results on transmission efficiency and throughput with those determined during the Phase I study. For block-by-block ARQ and continuous ARQ-Type I, the transmission efficiency (see equations (6-5) and (6-10)) varies inversely with R_{II}, the Class II transmission rate. Therefore, all else being equal, we would expect the efficiencies for the model to be greater than that obtained in Phase I since we employ a lower carrier rate (230.4 KBPS versus 1.544 MBPS in Phase I). This can be seen from the curves and from the chart of optimal efficiencies in Table 6-1. Specifically for a 10^{-3} BER, the optimum efficiency for block-by-block ARQ is .1617 and for continuous ARQ, Type I it is .2744. ⁊ Phase I these quantities were .0335 and .0658, respectively. The efficiency for continuous ARQ-Type II is not a function of Class II transmission rate and thus will not change as the carrier rate is varied. Therefore, the curves in Figures 6-3 and 6-6 are identical to those derived during Phase I.

Class II throughput is just the transmission efficiency multiplied by the transmission rate of Class II data. The transmission rate is about 20 percent of the 230.4 kBPS carrier rate, or 44.8 kBPS. This is consistent with Phase I (where 308,800 kBPS was the nominal data rate, or 20 percent of 1.544 MBPS rate) and allows for busy hour date transmission of two packets per frame, each 224 bits in length. The throughput, plotted with transmission efficiency along the vertical axis in Figures 6-1 through 6-6, varies from zero to a maximum of 44.8 kBPS, about one-seventh the maximum Class II throughput considered in Phase I. For example, the maximum throughput at a 10^{-3} bit error rate for block-by-block ARQ, continuous ARQ-Type I and continuous ARQ Type-II is 7.24 kBPS, 12.29 kBPS, and 20.92 kBPS, respectively (see Table 6-1). In Phase I the corresponding values of throughput were 10.34 kBPS, 20.32 kBPS, and 144.21 kBPS.

Table 6-3 summarizes the Class II throughput and its slope for the three types of ARQ protocol. In Figure 6-8 and 6-9 these are plotted as a function of the transmission rate for the case where $E_B = 10^{-3}$ and $P_{opt} = 224$ bits. As the transmission rate increases, the curves for block-by-block and continuous ARQ-Type I level off approaching values of 4.369 kBPS and 21.766 kBPS respectively, in the

TABLE 6-3. EQUATIONS OF CLASS II THROUGHPUT FOR VARIOUS ARQ PROTOCOL

ARQ PROTOCOL	CLASS II THROUGHPUT	SLOPE OF CLASS II THROUGHPUT
BLOCK-BY-BLOCK	$\frac{RIE^{-PE_B}(1-DC)}{I+C+WR(1-DC)}$	$\frac{I(I+C)E^{-PE_B}(1-DC)}{[I+C+WR(1-DC)]^2}$
CONTINUOUS — TYPE I	$\frac{RIE^{-PE_B}(1-DC)}{I+C+WR(1-DC)(1-E^{-PE_B})}$	$\frac{I(I+C)E^{-PE_B}(1-DC)}{[I+C+WR(1-DC)(1-E^{-PE_B})]^2}$
CONTINUOUS — TYPE II	$\frac{RIE^{-PE_B}(1-DC)}{I+C+P(1-DC)(1-E^{-PE_B})}$	$\frac{IE^{-PE_B}(1-DC)}{I+C+P(1-DC)(1-E^{-PE_B})}$

$E_B = .001$, $I = 164$, $P = 224$, $DC = .05$

7777-76E

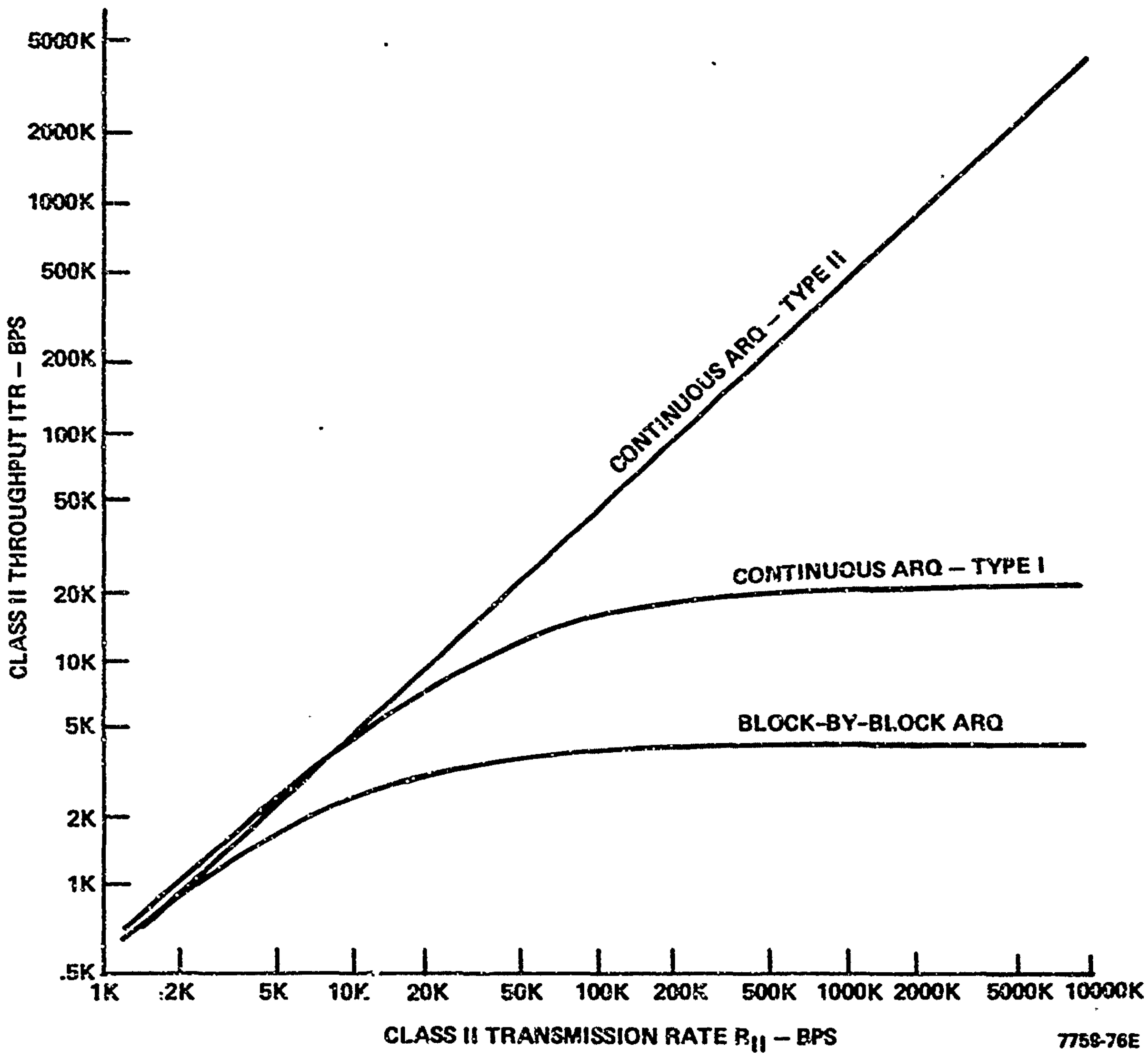


Figure 6-8. Variation of Class II Throughput with Transmission Rate

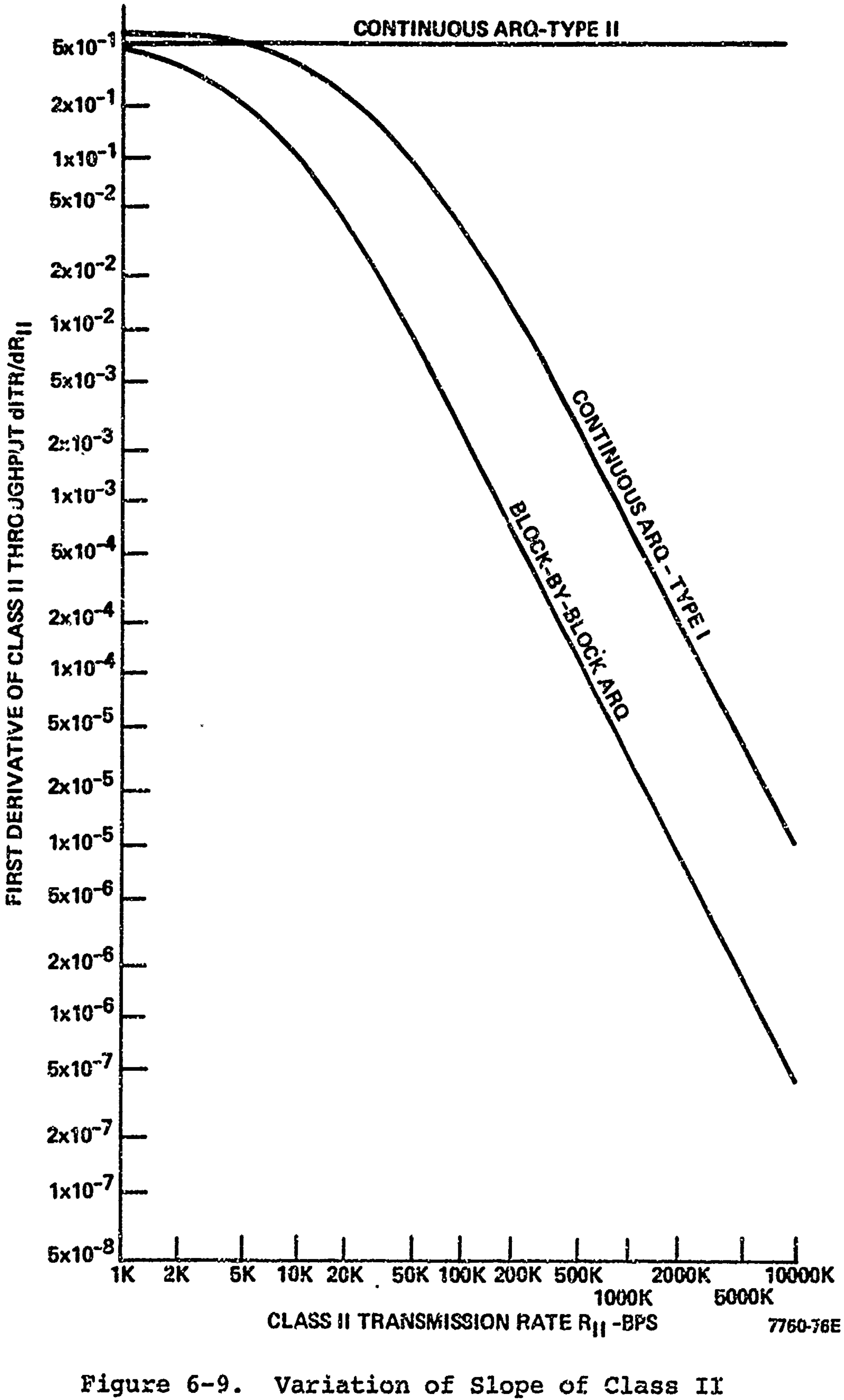


Figure 6-9. Variation of Slope of Class II Throughput with Transmission Rate

limit (see Table 6-3 and Figure 6-8). The throughput associated with continuous ARQ-Type II, however, keeps increasing in a manner proportional to the increase in transmission rate. At very high transmission rates repeating only the information in error is a highly efficient way of transmitting data and leads to high throughput capability*. Transmission capacity is not being wasted by waiting for data to be acknowledged before sending the next block or by repeating data that was sent correctly the first time. Our operating point for this investigation is at 44.8 kBPS. Here the advantage of continuous ARQ-Type II is not as pronounced as was the case of 308.8 kBPS, the operating point for Phase I. Notice that at extremely low transmission rates the advantage appears to shift away from Type I ARQ to the other protocols. However, this is a complicated function of the Class II transmission rate, the packet length specified, and the waiting time W and will vary considerably depending upon the values assumed for these parameters.

In Figure 6-9 the slope of the throughput at high transmission rates for block-by-block ARQ and continuous ARQ-Type I is inversely proportional to the square of the transmission rate. For continuous ARQ-Type II, this slope is a constant function of transmission rate.

For the experimental system, continuous ARQ-Type II protocol appears to have the best characteristics of throughput and efficiency for our transmission rates of interest. This will be further examined in the next section.

* It is assumed that a sufficiently fast processing capability is available to accomplish the required data transfer.

6.3 ANALYSIS OF BLOCKING AND DELAYS

6.3.1 Blocking Probability for Voice Calls and Expected Delay for Data Messages

Traditional parameters of performance for voice and data systems are the blocking probability for voice calls and the expected waiting time for data delivery. Analytical methods for computing these measures are well documented for voice and data systems taken individually. Here we are concerned with computational procedures for composite voice/data systems (in which interaction between the voice and data boundary is dynamically adjustable) and varies as a function of traffic demand by class.

It is necessary that computational procedures employed at this design stage of the investigation allow calculations to be made rapidly, yet provide sufficient accuracy for describing the performance characteristics of the model. Exact analytical solutions for system performance exist, [9]; however, they engender considerable computational complexity. For example, determining the blocking probability for voice calls requires the solution of a large system of linear equations while the waiting time for data requires the derivation of complex roots. Although the techniques are very accurate, the need for such powerful computational schemes, which require a large computer for implementation, is questionable for this immediate application.

During the Phase I literature review, various approximations to calculation of these performance measures were derived [9,13,21]. For blocking probability of voice or virtual circuit switched traffic (Class I), the erlang loss formula was used, while for the expected waiting time, an approximation was suggested based on substituting the M/M/N queueing model for the more complicated M/D/N system. A review of the derivations of these formulae with respect to our application and some preliminary calculations, showed that the approximations introduce errors which are small compared with the 10 msec frame interval. The erlang loss formula, given by

$$P_L = \frac{(E_I)^S / S!}{\sum_{j=0}^{S} (E_I)^j / j!} \qquad (6\text{-}16)$$

is the traditional method used to calculate blocking probability and is justified when the expected number of voice arrivals (λb) is small. E_I is the Class I traffic offered to the trunk and equals λ/μ erlangs, where λ is the call origination rate, $1/\mu$ is the holding time per call, $b = 10$ msec is the service time of the system, and S is the finite number of voice channels available.

An exact solution for the expected delay involves the M/D/N queueing model, which specifies Poisson arrivals and a constant service time. However, these conditions yield extremely complex equations whose solution becomes tractable only by using a large computer. Therefore, the M/M/N model, which specifies Poisson arrivals and exponentially distributed service times, was substituted for the more exact M/D/N model to reduce the complexity of the calculations. The M/M/N approximation yields a larger average waiting time than the M/D/N model and therefore, errors, introduced by the approximation, are such as to produce conservative results.

In the expression for delay, EW is given by

$$EW = b + \frac{b}{2} + \frac{Nb.E(N,\rho_2)}{(N-\rho_2)\ [N - \rho_2 + \rho_2\ E(N,\rho_2)]} \qquad (6\text{-}17)$$

where $\frac{b}{2}$ is the average waiting time to get into the queue to be serviced,

$$\frac{NbE\ (N,\rho_2)}{(N-\rho_2)\ [N - \rho_2 + \rho_2 E\ (N,\rho_2)]}$$

is the time spent waiting in the queue, and b is the service time once entry into the system is achieved. N is the maximum amount of data allowed in the system, while ρ_2 represents the offered data load. Both the M/M/N and M/D/N models are examples of imbedded Markon chains, where, at entry points denoted by the fixed time interval b, the system services entries waiting in the queue.

Equations (6-16) and (6-17) compute the "steady state" values of the blocking probability and the expected delay time. In practical networks, however, the traffic may vary from very small trunk utilization to periods of time during which the traffic load exceeds the available trunk capacity. During these periods of overload, the queue size and delay of data packets can grow without limit, since Class II data is not blocked, but only delayed. If an overload condition were to last a time approaching infinity, the queues and the queueing delays would also grow towards infinity. Our steady state solutions disregard this overload effect and as a result queue size and average waiting time may be underestimated by not taking into account the time spent in the queue during overload. Kummerle [11] suggested a correction term based on calculating approximately the momentary mean queueing delays corresponding to each overload state and using the respective probabilities of these states in a weighted sum of the delays. The corrective term was then added to the steady state values of expected delay in order to account for this overload phenomenon. Kummerle's results, however, even with the corrective term, show the delay and queue sizes to be underestimated by factors as high as 2 to 1 when the data traffic load exceeds the trunk capacity exclusively reserved for data. Therefore, until we are in a position to more accurately determine the peak queue distributions that would develop in an experimental SENET-DAX network, we will continue to use the steady state estimates of delays and queue size that were developed in the Phase I study. We must be careful, however, to recognize that calculations of waiting time or queue length are only approximate, particularly for high traffic utilization.

6.3.2 Variation of Waiting Time and Trunk Utilization with Traffic Load

The steady-state approach described above was modified to handle movable domain boundaries and variable length channels for voice and data packets. With these computational tools then defined, a parametric analysis of delay versus data service was performed for the experimental system. An analysis was made of a 230.4 kBPS trunk used to carry a Class I voice load of 4 erlangs and a parametric data load varying from 1 to 6 data packets, each 224 bits in length. The voice load was chosen to approximate the typical peak-hour voice traffic expected on an interswitch trunk of the experimental network. Expected holding time per call was 5 minutes. A frame interval of 10 msec was assumed.

The 2304 bits of the 230.4 kBPS trunk were allocated to voice channels, each 155 bits in length, data packets of 224 bits each, one 232-bit CCIS message, a 16-bit SOF marker, and 10 bits set aside for bit stuffing (via ADCCP protocol). The 155 bits per voice channel were obtained from an average distribution of voice terminals served by the SENET-DAX model and is based on 1985 DOD projected statistics. Each voice user is allocated a normalized 155-bit voice channel; thus, 4 erlangs of voice occupies 4 channels, or 620 bits. Packet-switched data traffic occupies the remaining Class I/Class II space not used by voice. One 232-bit channel was reserved for CCIS traffic for establishing and breaking down calls, and for dynamic control of trunk allocation.

Waiting time delays and trunk utilization factors were computed for both fixed and flexible Class I/Class II boundaries (see Table 6-4). Probability of loss (voice call blocking probability) was considered independent of data traffic (priority is not considered here). Four erlangs of voice traffic offered to about 80 percent of the total frame interval produced a probability of lost calls due to trunk blockage of .001 (0.1 percent). However, any unused voice

TABLE 6-4. VARIATION OF THE WAITING TIME AND DATA UTILIZATION WITH OFFERED DATA TRAFFIC FOR FIXED AND MOVABLE BOUNDARY CASES; 230.4 KBPS TRANSMISSION RATE

DATA PACKETS OFFERED	PROL. OF BLOCKING (4 ERLANGS) (VOICE)	VOICE UTILIZATION	FIXED BOUNDARY		MOVABLE BOUNDARY		TOTAL TRUNK UTILIZATION
			WAITING TIME (MSEC)	DATA UTILIZATION	WAITING TIME (MSEC)	DATA UTILIZATION	
1	.1%	0.34	∞	–	15.0012	.158	.478
2	.1%			–	15.045	.316	.576
3				–	15.33	.473	.673
4				–	16.87	.631	.770
5				–	20.81	.789	.867
6	.1%	0.34	∞	–	∞	–	–
7	–	–	∞	–	∞	–	–

7746-76E

channels can be used for packet-switched data. The voice utilization is 34 percent assuming an offered load of 4 erlangs and taking into account calls which are blocked.

The data load was varied parametrically from 1 to 6 packets. Waiting time consists mainly of the fixed component (15 msec), which represents the time spent waiting to enter the system (on the average one-half of the frame interval), and the service time (equal to the frame period). An additional delay is present due to expected waiting time in the input queue and is variable. Trunk utilization is much better for the flexible boundary case, where data is allowed to make use of all available frame capacity, than for the fixed boundary case where data is restricted to one packet each frame. For the flexible boundary case, as data load is increased, expected waiting time is relatively constant, up to about 5 packets (80 percent of the available data capacity). After this, waiting time increases rapidly. Up until this point, delay consists mainly of the fixed component (1.5) multiplied by the service time. Excessive queueing delay comes in above the 80 percent data utilization.

For the flexible boundary case, 4 erlangs of voice traffic, at a grade-of-service of .1 percent can be accommodated along with 2 data packets with insignificant delay (less than .1 msec), on a single 230.4 kBPS link. This represents a data utilization of 31.6 percent and an overall trunk utilization of 57.6 percent. Alternatively 5 packets per frame can be accommodated with 20.8 msec delay. A corresponding tradeoff analysis cannot be made for the fixed boundary case, since no more than 1 packet per frame can ever be sent on the link.

Table 6-5 shows the same information for a trunk rate of 1.544 MBPS (T1 carrier) with normalized voice channels and 224-bit data packets as proposed for the experimental system. At this transmission rate, for the moving boundary case, 63.5 erlangs of voice traffic can be carried at a .1 percent grade-of-service along with 15 data packets with negligible delay. This represents a data utilization of 63.5 percent and an overall trunk utilization of 87.4 percent. Up to 20 packets can be carried each frame with 16.2 msec delay. Comparison of these results with those in Table 6-4 shows that there is a loss in overall service capability that is less than the proportionate reduction in bit rate in going from a T1 carrier rate to the experimental system transmission rate of 230.4 kBPS.

6.3.3 The Effect of a Noisy Environment on the Data Traffic Load

Data traffic load will increase in a noisy environment due to the need for ARQ error control. This effect has been examined in paragraph 6.2, where a mathematical model of the effect of noise on transmission efficiency and throughput was investigated. Three types of ARQ error control protocol were considered: Block-by-Block, Continuous-Type I and Continuous-Type II. Examination of the curves in paragraph 6.2 showed that continuous ARQ-Type II, where only the packet in error is retransmitted, is the most efficient of the three

TABLE 6-5. VARIATION OF THE WAITING TIME AND DATA UTILIZATION WITH OFFERED DATA TRAFFIC FOR FIXED AND MOVABLE BOUNDARY CASES; T1 CARRIER RATE

DATA PACKETS OFFERED	PROB. OF BLOCKING (63.5 ERLANGS VOICE)	VOICE UTILIZATION	FIXED BOUNDARY		MOVABLE BOUNDARY		TOTAL TRUNK UTILIZATION
			WAITING TIME (MSEC)	DATA UTILIZATION	WAITING TIME (MSEC)	DATA UTILIZATION	
1	.1%	.89858	15.0093	.178	15.0000	.042	.671
2			15.1900	.359	15.0000	.085	.686
3			16.1803	.534	15.0000	.127	.700
4			20.5555	.713	15.0000	.169	.715
5			∞	–	15.0000	.212	.729
6			∞	–	15.0000	.254	.744
7			∞	–	15.0000	.296	.758
10			∞	–	15.0002	.424	.802
15			∞	–	15.0475	.635	.874
20			∞	–	16.2050	.847	.947
23	.1%	.89858	∞	–	∞	–	–
25	–	–	∞	–	∞	–	–

7780-76E

forms of protocol. The other procedures require retransmission of more data than does continuous ARQ-Type II for each error occurrence. This increases occupancy time in the transmit queue and thus requires larger queues (more memory) on a per link basis. There is also the increased complexity associated with continuous ARQ-Type I where validly received packets which have been processed must not be processed a second time.

For a bit error rate of 10^{-3}, the optimum packet size associated with block-by-block ARQ is P_{opt} = 728 bits and the corresponding efficiency is 16.17 percent (see Table 6-1). The busy hour data load is 448 bits per frame (approximately 20 percent of the total frame size). The peak equivalent data load in this noise environment is $\frac{448}{.1617} = 2771$ bits. This means that because of the inefficiency of data transfer in this high noise environment, it would take more bits than are available for the entire frame to transmit the two data packets. Since this exceeds the frame capacity, an overflow condition would exist resulting in an infinite queue.

Under the same conditions, i.e., a bit error rate of 10^{-3}, the optimal packet size for Continuous ARQ-Type I is P_{opt} = 314 bits and the efficiency is 27.44 percent. The peak data load for this case is $\frac{448}{.2744} = 1633$ bits. This still results in an overload condition because, besides the two packets, the frame is also required to accommodate during the busy hour, four erlangs of voice, an SOF marker, one CCIS message and 10 bits for bit stuffing. These results tend to indicate the impracticality of using block-by-block or Continuous Type I ARQ over a noisy wideband link.

6.4 SYSTEM SIZE AND TRAFFIC HANDLING CAPABILITY

In a conventional switching system sizing is usually defined by the quantities of lines, trunks, and service connections accommodated by the switch, for a specified level of traffic. In the SENET-DAX concept, however, various classes of traffic are accommodated, each of which are composed of different transmission rates. This kind of system is more appropriately sized in terms of throughput, with the trunk transmission rate providing the basic unit of throughput and the corresponding system size that generates that throughput providing the unit of system expansion.

In order to calculate the number of voice and data subscribers served by the model, the following assumptions, based on 1985 DOD projected statistics, [26], are made for voice traffic:

a. An average holding time of 3 minutes for local calls and 5 minutes for trunk calls

b. A call distribution by transmission rate given by:

10 percent at 2400 BPS	(vocoder)
10 percent at 4000 BPS	(LPC)
15 percent at 8000 BPS	(APC)
50 percent at 16,000 BPS	(CVSD)
10 percent at 32,000 BPS	(CVSD)
5 percent at 50,000 BPS	(KY3)

c. All calls are full-duplex.

Based on these assumptions, the effective weighted transmission rate is 15,540 BPS and the average number of bits in a frame per voice call is 155.4. Assume that 80 percent of the frame is used to carry voice traffic. Then 4 erlangs of voice traffic with a grade-of-service of

0.1 percent, 1 CCIS message, and 2 data packets 224 bits in length with insignificant delay can be accommodated during the busy hour on a single 230.4 kBPS link.

The number of voice subscribers that generate a trunk traffic of 4 erlangs can be determined from the expression

$$N_V = \frac{A}{a} \qquad (6\text{-}18)$$

where A is the total two-way traffic in erlangs for the N voice subscribers and a is the two-way traffic in erlangs for each subscriber. Assuming a distribution of two-thirds of the traffic carried over the trunk, i.e., line-to-trunk and trunk-to-line; and the rest carried as local traffic, N_V is calculated as follows:

$$\frac{2A}{3} = 4 \text{ erlangs}$$

and $A = 6$ erlangs.

Given $a = .3$ erlangs

$$N_V = \frac{6}{.3} = 20 \text{ voice subscribers}$$

The number of data subscribers that can be supported is equal to the total data traffic on the link divided by the average data traffic per subscriber terminal. The data trunk traffic is 44.8 kPBS. Assuming that this is two-thirds of the total data traffic on the link, the total becomes 1.5 x 44,800 bits/sec. Let

N_D = number of data terminals

λ = total number of two-way messages per busy hour

ℓ = average message length (bits)

Then

$$N_D = \frac{(1.5 \text{ x } 44,\ 800)\ (3600)}{\lambda \ell} \qquad (6\text{-}19)$$

From reference [26] λ = 24 messages per busy hour, and = 28,000 bits per message. Therefore

$$N_D = \frac{(1.5 \text{ X } 44,800)(3600)}{(24)(28000)} = 360 \text{ data subscribers.}$$

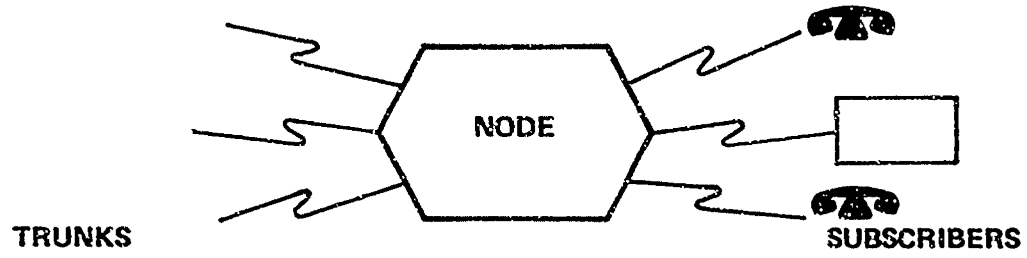
Thus the single 230.4 kBPS link has the capacity to support up to 20 voice subscribers and 360 data subscribers.* Since the minimum system size can be considered in terms of one 230.4 kBPS trunk, where the link has been defined as the ability to carry local traffic generated according to the assumptions made earlier, a single switch can handle a maximum of 380 subscribers. The system architecture, however, is flexible enough to allow a lesser subscriber population and a higher concentration of (partially used) 230.4 kBPS trunks to other nodes.

The maximum size of the model has been designed to accommodate up to five 230.4 kBPS trunks (see Section 3). In Section 2 it was shown that this requires a memory access cycle time of 248 nanoseconds, assuming typical program complexity (see Figure 2-1, distributed curves, Ap = 15 access/byte). Since state-of-the-art cycle times for solid state memories are in the order of 100 nsec, our requirements are well within the range necessary for available, off-the-shelf hardware.

* The calculation of these quantities did not take into account either the movable boundary or the increase in data resulting from ARQ in a noisy environment.

Traffic handling characteristics of the experimental system are tabulated in Table 6-6. It is assumed that each node services three 230.4 kBPS trunks. Upper bound calculations for Class I assume that the entire frame is reserved for voice except for the SOF flag and enough capacity to send one CCIS message. Upper bound calculations for Class II data packets assume that 2 packets can be processed per link during each frame with less than .1 ms queueing delay.

TABLE 6-6. SYSTEM TRAFFIC HANDLING CHARACTERISTICS



		TRUNK CALLS		
		PER LINK	PER NODE	LOCAL CALLS
CLASS I (VOICE)	BUSY HOUR	.0133 CALLS/SEC	.04 CALLS/SEC	.02 CALLS/SEC
	UPPER BOUND	.05 CALLS/SEC	.14 CALLS/SEC	.07 CALLS/SEC
CLASS II (DATA)	BUSY HOUR	200 PACKETS/SEC	800 PACKETS/SEC	–
	UPPER BOUND	500 PACKETS/SEC	2000 PACKETS/SEC	–

ASSUMPTIONS:

- BUSY HOUR TRAFFIC OF 4 ERLANGS OF VOICE AT 0.1% GRADE-OF-SERVICE AND 2 DATA PACKETS 224 BITS IN LENGTH
- FRAME INTERVAL = 10 MSEC
- CALL DURATION = 5 MINUTES
- THREE LINKS PER NODE

7768-76E

6.5 CROSS-OFFICE AND CROSS-NETWORK DELAYS FOR TYPICAL VOICE AND DATA TRAFFIC

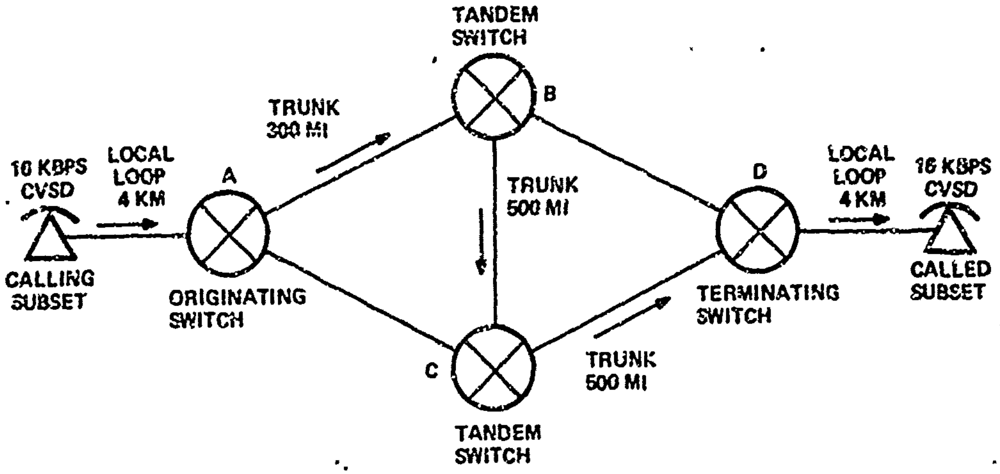
Cross-office and cross-network delays are important measures of any switched network and depend upon the topology of the network and the processing capability of each of the constituent switching nodes. For the experimental SENET-DAX network, operational limits on delay can be classified according to the distinct classes of traffic which the network will be required to carry. Class I terminals operate synchronously such that once the information flow is established, it continues for the duration of the call. Typical examples are voice and facsimile. These calls remain unaffected as long as cross-network delays are less than about 250 msec. Class II interactive TTY and query/response terminals are conveniently handled by packet-switching techniques. These calls generally require delays on the order of a second or less for responsive operation. Another type of Class II call is bulk data transfer, typified by computer-to-computer calls and the Autoden I type of record traffic. Here time is not a critical factor. The message can be delivered by the network on a space-available basis.

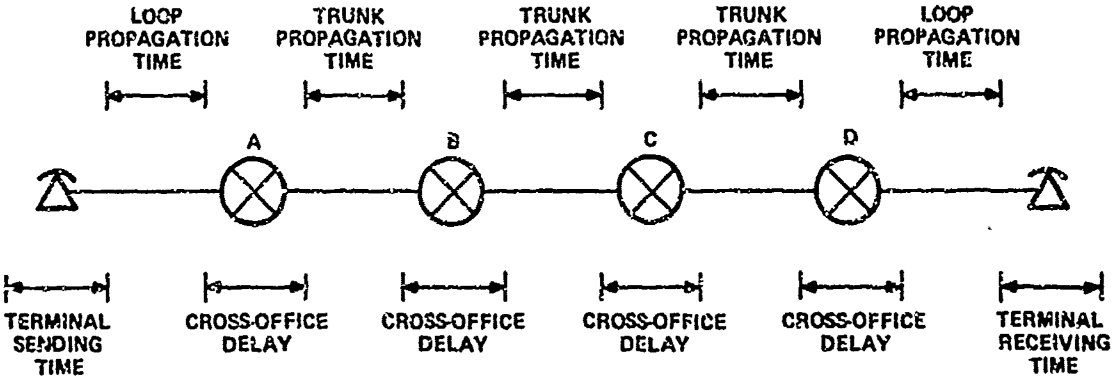
In this section, cross-office and cross-network delays will be determined for typical voice and data calls on the experimental SENET-DAX network. The examples to be considered include a secure 16 kBPS Class I voice call; an interactive packet-switched data call between a terminal and computer; and a CCIS message sequence used for call setup. In each example the component elements in the signal path between the transmitting and receiving terminals that contribute to delay are identified and the delay estimated. These delays are then accumulated and the cross-network delay for each type of call is specified.

6.5.1 Cross-Network Delay for 16 kBPS Voice Call

Figure 6-10 shows the network connections associated with a 16 kBPS secure voice conversation. The circuit, which consists of three trunks and two subscriber loops, was chosen to typify the worst-case delay conditions associated with the 4-node experimental network. This path connection was also used to demonstrate other examples of network calls, as will become evident in the paragraphs which follow. In this example we assume that the virtual circuit has been established between subscriber subsets along the path shown and will be maintained for a call duration of 5 minutes. Delay times associated with the component elements along this path connection are given in Figure 6-10.

The total network delay consists of the delays in the sending and receiving terminals, the propagation delay over the two 4-kilometer subscriber terminal loops and the three 500-mile trunks, and the cross-office delays across each of the four SENET-DAX switches. The subsets are digital secure voice terminals, such as the DVST, and use 16 kBPS continuous variable slope delta (CVSD) modulation for analog-to-digital conversion. The digital output is converted to quasi-analog form by a conditioned diphase modem for transmission over the loop to the switch. Figure 6-11A shows the delays associated with typical functions of the digital secure voice terminals. Individual





	CALLING SUBSET	LOCAL LOOP	ORIGINATING SWITCH	TRUNK CIRCUIT	TANDEM SWITCH	TRUNK CIRCUIT	TANDEM SWITCH	TRUNK CIRCUIT	TERMINATING SWITCH	LOCAL LOOP	CALLED SUBSET
TERMINAL SENDING TIME	.4375										
TERMINAL RECEIVING TIME											.4375
LOOP PROPAGATION TIME		.021								.021	
TRUNK/LOOP RECEIVING TIME			.4688		.4688		.4688		.4688		
PROCESSOR DELAY			16.0		15.0		15.0		16.0		
TRUNK/LOOP SENDING TIME			.911		.911		.911		.911		
TRUNK PROPAGATION TIME				4.25		4.25		4.25			

dimensions in msec

AVERAGE CROSS-NETWORK DELAY = 81.19 MSEC
CROSS-NETWORK DELAY (90% CONFIDENCE LEVEL) = 88.70 MSEC
CROSS-NETWORK DELAY (99.7% CONFIDENCE LEVEL) = 98.53 MSEC

7774-76E

Figure 6-10. Cross-Network Delay for 16kBPS Voice Call

blocks were described in the Phase I Final Report [26] and estimates of delay were made. The terminal delay from the time that the first bit of the signal enters the output buffer until it is emitted by the modem and propagates down the loop is 7 bit times. This corresponds to a sending and receiving terminal delay of

$$T_{SL} = \frac{7 \text{ bits}}{16 \text{ kBPS}} \approx .4375 \text{ msec.}$$

The propagation delay of trunks and loops are calculated according to the formula

$$T_P = \sqrt{\varepsilon \ (L)/c} \qquad (6\text{-}20)$$

where L is the path length, ε is the dielectric constant of the medium, and c is the velocity of light (3 x 10^8 meters/second). The propagation delay is defined as the time required for a signal to travel down a sending or receiving trunk or loop. This time is less for microwave radio or satellite than it is for coaxial cable. However, in order to "upper bound" all propagation delays calculated for the experimental system, trunks will be considered to be coaxial cable and loops will be twisted-pair field wire, such as WF-16. For these conditions, the 500 mile trunk propagation delay is TP_{TR} = 4.25 msec and the 4 kilometer loop propagation delay is T_{PL} = .021 msec.

The functional components which contribute to cross-office delay at the SENET-DAX node are shown in Figure 6-11B. This figure illustrates the interface between the switch and the 230.4 kBPS trunk circuit and between the switch and the 16 kBPS loop. The cross-office delay is given by

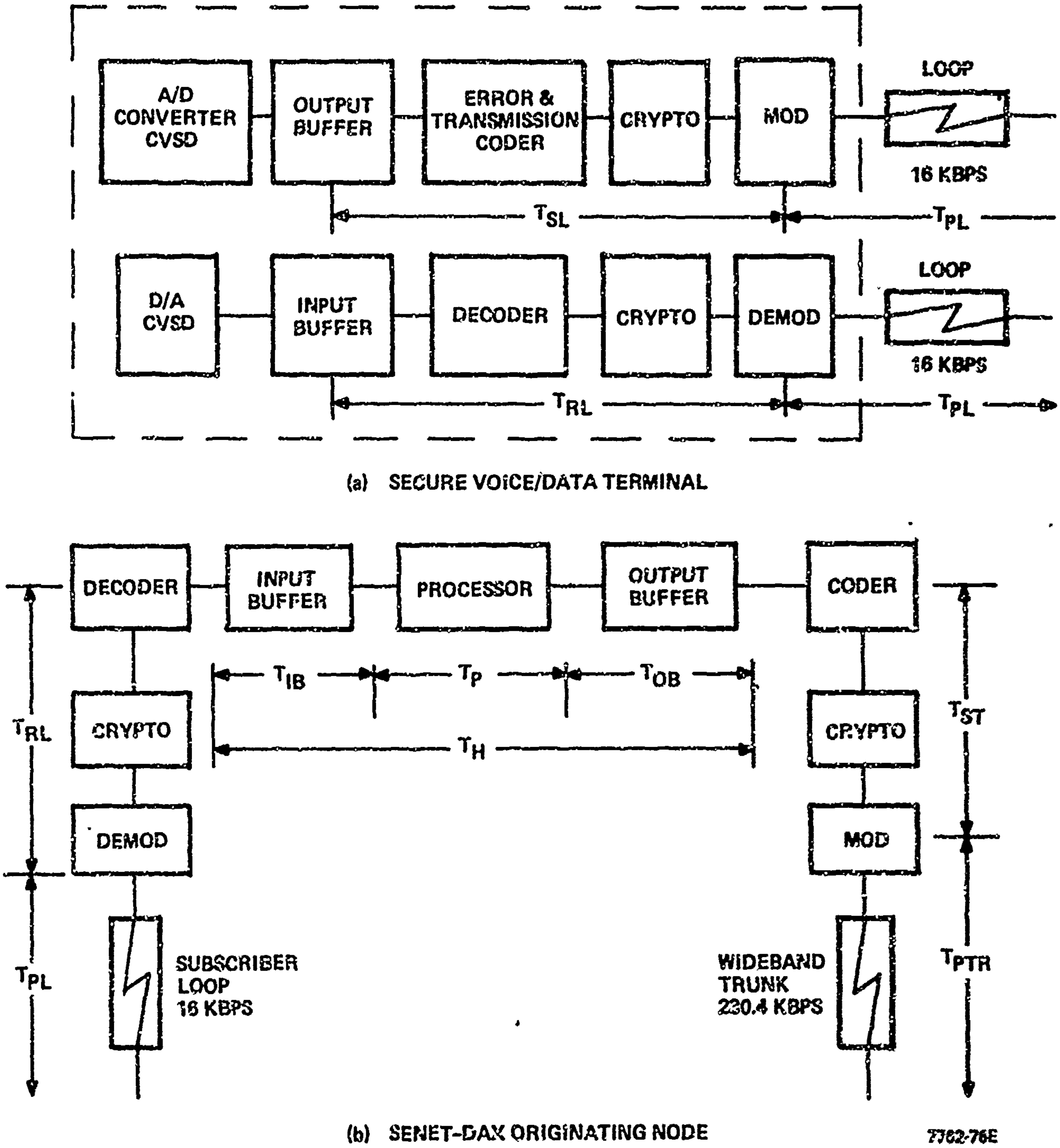


Figure 6-11. Signal Delay Time Diagrams

$$T_C = T_{RL} + T_H + T_{ST} \qquad (6\text{-}21)$$

and is the sum of the delays in the receiving and sending interface units and the processing delay of the switch. This figure shows an originating switch where the receiving interface is a loop and the sending interface is a trunk. For tandem nodes both the receiving and sending interfaces are trunks.

The receiving loop interface delay consists of the delay in the modem, crypto and the decoder/line termination unit and is assumed to be 6 bit-times. The signaling delay of 1 bit for CVSD is included in the sending interface delay. Assuming an average polling delay in the Decoder/LTU of one-half of the load period at a 16 kBPS rate, the value of TRL is estimated to be .4688 msec. T_{ST}, the transmitting trunk interface delay, consists of the delay in the modem, crypto, and coder blocks plus the time necessary to transmit the 160 bits of the voice channel onto the line. Assuming 6 bit-times of delay, or 4 microseconds, for the modem, crypto, and coder, the value of T_{ST} for a 16 kBPS voice channel in a 230.4 kBPS trunk is 911 microseconds. Figure 6-12 shows the detailed processing functions of the Coder and Decoder in terms of the link processing structure of the experimental system.

The processing delay is given by

$$T_H = T_{OB} + T_P + T_{IB} \qquad (6\text{-}22)$$

and consists of input and output buffer delays and the delay due to processing the 160 voice bits per frame. No queues will be formed for

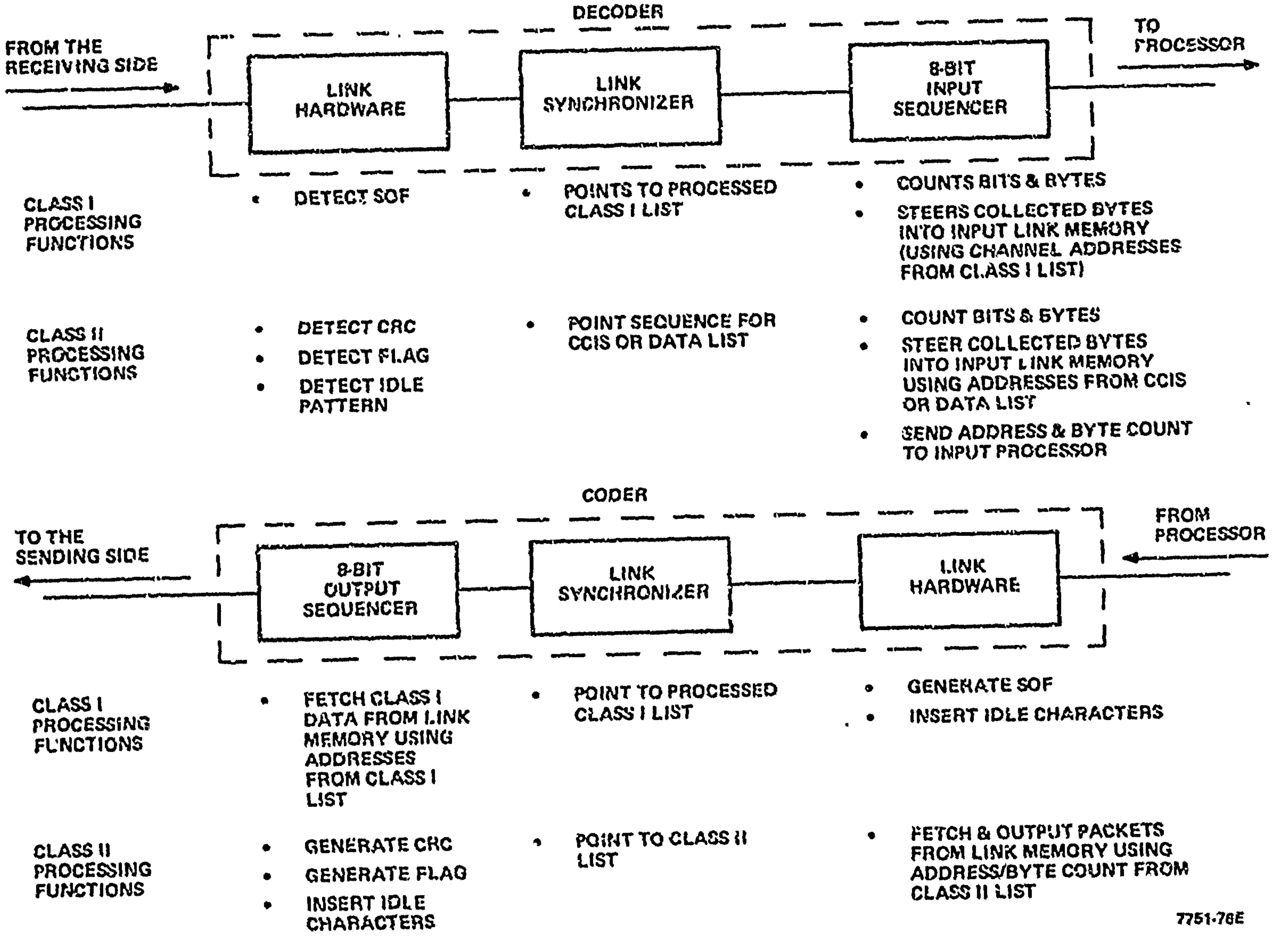

DECODER
FROM THE RECEIVING SIDE
LINK HARDWARE
LINK SYNCHRONIZER
8-BIT INPUT SEQUENCER
TO PROCESSOR
CLASS I PROCESSING FUNCTIONS
• DETECT SOF
• POINTS TO PROCESSED CLASS I LIST
• COUNTS BITS & BYTES
• STEERS COLLECTED BYTES INTO INPUT LINK MEMORY (USING CHANNEL ADDRESSES FROM CLASS I LIST)
CLASS II PROCESSING FUNCTIONS
• DETECT CRC
• DETECT FLAG
• DETECT IDLE PATTERN
• POINT SEQUENCE FOR CCIS OR DATA LIST
• COUNT BITS & BYTES
• STEER COLLECTED BYTES INTO INPUT LINK MEMORY USING ADDRESSES FROM CCIS OR DATA LIST
• SEND ADDRESS & BYTE COUNT TO INPUT PROCESSOR
CODER
TO THE SENDING SIDE
8-BIT OUTPUT SEQUENCER
LINK SYNCHRONIZER
LINK HARDWARE
FROM PROCESSOR
CLASS I PROCESSING FUNCTIONS
• FETCH CLASS I DATA FROM LINK MEMORY USING ADDRESSES FROM CLASS I LIST
• POINT TO PROCESSED CLASS I LIST
• GENERATE SOF
• INSERT IDLE CHARACTERS
CLASS II PROCESSING FUNCTIONS
• GENERATE CRC
• GENERATE FLAG
• INSERT IDLE CHARACTERS
• POINT TO CLASS II LIST
• FETCH & OUTPUT PACKETS FROM LINK MEMORY USING ADDRESS/BYTE COUNT FROM CLASS II LIST
7751-76E

Figure 6-12. Decoder/Coder Functional Processing

Class I calls during overload since any surplus calls are blocked. Therefore, the only delays for Class I traffic will be the polling delays in the input and output buffers.

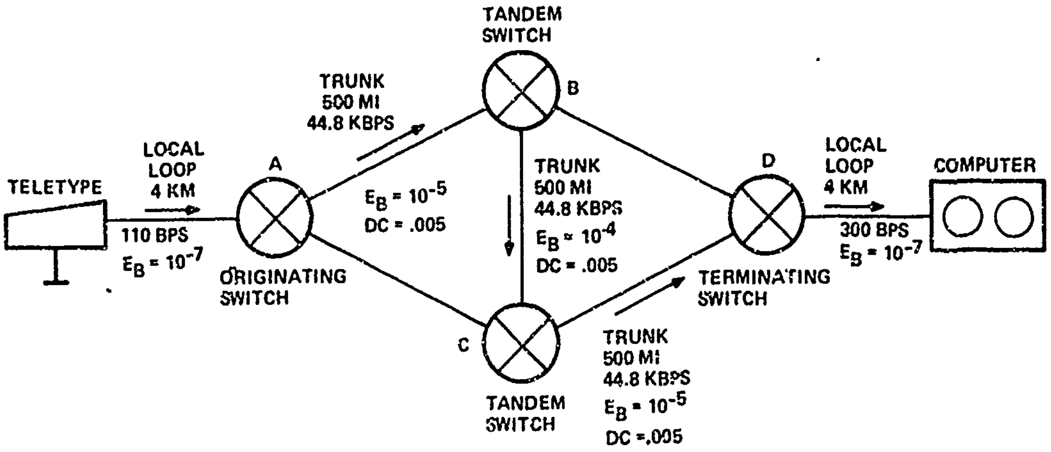
The input processor will effect transfer of the 160 bits per frame to the input buffer. This information is accumulated until a complete 160 bit segment is received. Assuming a 230.4 kBPS data rate for the trunk, the entire 160 bits will be transmitted in 0.69 msec starting at some random point in the Class I region of the 10 msec frame. This position will change during the course of the conversation as calls are completed and the Class I region is recompacted. The delay in transmitting the first bit on the output trunk is random, varying from 0 to a complete 10 msec frame. The average delay is one-half the frame interval, or 5 msec. Thus the total processing delay will consist of the 10 msec frame interval plus this stochastic component, for a total of 15 msec per node. The range will vary from 10 to 20 msec, with a variance per node of .833 msec (assuming a uniform distribution for the random component). The average cross-office delay of a node becomes T_C = 15 + .4688 + .911 = 16.38 msec.

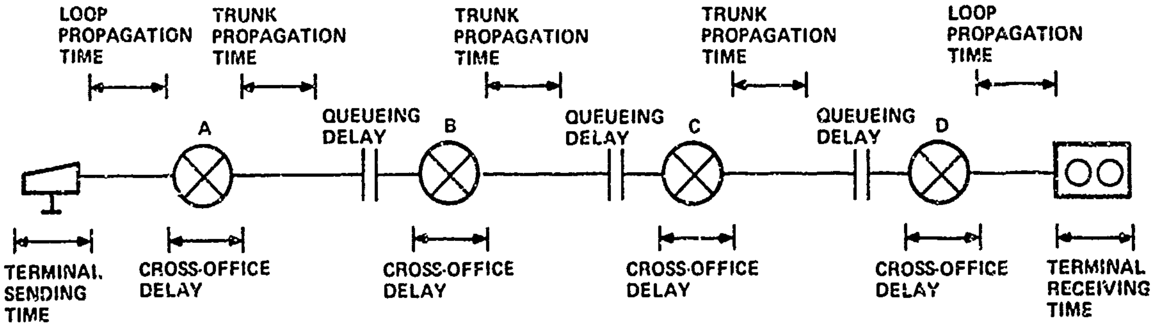
For the case of end nodes, an additional millisecond is included to account for buffering capacity necessary to allow for propagation delay variations and clock drifts during fades. The additional time is arbitrarily assigned to T_{OB} and would allow for variations up to 16 bits during a fade or due to propagation anomalies. This adjustment is only necessary at the originating and terminating nodes.

The resultant cross-network delay after the connection has been established is 81.19 msec. Of this total, 12.79 msec represents the propagation delay over the 1500-mile circuit, .875 represents the total of the transmitting and receiving terminals, and 67.52 represents the cross-network delays of the switches (34.76 msec for the two terminal nodes and 32.76 for the two tandem nodes). The only stochastic element in the delay is the half-frame expected value of delay from the time the block of 160 bits is received until it is inserted in its slot in an output trunk. Assuming this delay to be uniform over the frame interval and the cross-network delay to be normally distributed around 81.19 msec, the cross-network delay for the 1500-mile voice call would be 88.7 msec at a 90 percent confidence level and 98.53 msec at a 99.7 percent confidence level.

6.5.2 Cross-Network Delay for Packet-Switched Data

In this example an interactive data call between a 110 BPS teletype and a computer is sent over the network as packet-switched Class II traffic. Figure 6-13 shows the path connection for the call. A terminal message length of 2000 bits is assumed. The message is transmitted over a circuit which includes two 4-kilometer subscriber loops and three 500-mile trunk links. For the purposes of the experiment, the computer may be emulated by substituting the nodal processor at one of the nodes in place of a teletype terminal at the interface to a Class II access processor.





	TERMINAL	LOCAL LOOP	ORIGINATING SWITCH	TRUNK CIRCUIT	QUEUE	TANDEM SWITCH	TRUNK CIRCUIT	QUEUE	TANDEM SWITCH	TRUNK CIRCUIT	QUEUE	TERMINATING SWITCH	LOCAL LOOP	COMPUTER
TERMINAL SENDING TIME	10,240 (msg)													
TERMINAL RECEIVING TIME														21.6 (msg)
LOOP PROPAGATION TIME		.0[illegible]											.021	
TRUNK/LOOP RECEIVING TIME		59.06 (msg)				.145			.145			.145		
PROCESSOR DELAY		–	21.0			20.0			20.0			21.0		
TRUNK/LOOP SENDING TIME			5.134			5.134			5.134				6690 (msg)	
TRUNK PROPAGATION TIME				4.25			4.25			4.25				
QUEUEING DELAY					2.54			2.54			2.54			

dimensions in msec

CROSS-NETWORK DELAY (SINGLE DATA PACKET) = 118.207 MSEC
CROSS-NETWORK DELAY (ENTIRE DATA MESSAGE) = 268.21 MSEC
TOTAL CROSS-NETWORK DELAY (NETWORK DELAY PLUS
LOOP AND TERMINAL DELAYS) = 25.278 SEC

7775-76E

Figure 6-13. Cross-Network Delay for Interactive Data Call

A distribution of 80 percent for voice traffic and 20 percent for data traffic is assumed. This provides an effective Class II transmission rate of 44.8 kBPS on the trunks. Each link is 500 miles long and has a propagation delay of 4.25 msec. Bit error rates on the link are 10^{-5} and 10^{-4} except during bursts of 0.5 percent duty cycle, during which time the error rate is so high as to make the trunk useless for transmission. Noise bursts last from 2.5 to 50 msec and are spaced from 50 msec to one second. Loops are 4 km in length and have propagation delays of .021 msec. Loop error rates are 10^{-7} with a burst duty cycle of .0005. Peak hour voice load is 4 erlangs. Peak hour data load is 200 packets/second, packets are 224 bits in length, and Continuous ARQ-Type II protocol is used for error control.

The entire 2000-bit message is first transmitted to originating Node A, where it is broken up and formed into packets.* Packets are then transferred across the network to the terminating switch D. Packets transmit the network independently of each other. At the terminating node D, the message is reformed and delivered to its destination. Node D keeps track of all packets in a multipacket message and will not send data to the computer until all packets are in the output buffer.

*Actually, packetizing can take place before the entire message is received at A. A buffer is set up at A, and as this becomes filled, the contents are packetized and transferred along the network. However, because the optimum size of this buffer has yet to be determined, and because the delays due to the network portion of the path are the same for both cases, we will assume, for convenience, receipt of a complete message prior to packetizing.

Data packets will have an information field variable up to 164 bits per packet (60 bits are used for overhead) and continuous ARQ error control, with repeat of only the packet in error. The 2000-bit message will therefore be transmitted as 13 packets, all except the last having 224 bits. However, the number of packets which have to be transmitted in order to insure that 13 are received correctly is a function of packet length and bit error rate. This is described by the negative binomial law and by

$$\bar{N}_e = \frac{rq}{p} + r \qquad (6\text{-}23)$$

$$\sigma = \sqrt{\frac{rq}{p}}$$

where $\bar{N}_e$ is composed of r, the number of trials required to achieve the rth success plus the number of failures encountered before the rth success is met, and σ is the standard deviation. For 99.7 percent confidence, 16 packets would have to be sent to be confident that 13 packets were received correctly.

The input queueing delay at a node was estimated by the Pollaczek-Kintchin formula

$$Tqi = \frac{\rho \cdot \frac{1}{\mu}}{2(1 - \rho)} \left\{ 1 + \frac{\sigma_s^2}{(1/\mu)^2} \right\} \qquad (6\text{-}24)$$

where ρ is the utilization factor, 1/u is the service time, and O_s^2 is the variance of the service time. The maximum packet rate which the switch will ever see on one link will be 500 packets/second or 2 msec per packet. Assuming a processor utilization factor of .7, the queueing delay becomes T_{qi} = 2.543 ms per queue.

After the packet is processed at a node during the 10 msec incoming frame, it is placed at some position in the Class II region of the next 10 msec outgoing frame. This position is random but is most probably in the latter half of the frame because of Class I traffic. In the absence of Class I traffic the packet would directly follow the SOF marker. However, for a 90 percent voice/10 percent data distribution and a full 10 msec frame, the packet would appear in the last 10 to 12 percent of the frame. Therefore, we will neglect the stochastic variation and, as an upper bound, consider T_H, the processor delay, to be 20 msec.

T_{ST} and T_{RT} are computed to be $T_{ST} = \frac{224 + 6}{44.8k} = 5.134$ msec

and $T_{RT} = \frac{6 + .5}{44.8K} = .145$ msec. respectively.

It takes 18.24 seconds to transmit the 2000-bit message to node A via the 110 baud line and 6.69 seconds to transmit it from terminating node D to the computer. Other delays are calculated as for the 16 kBPS voice call in paragraph 6.5.1.

As shown in Figure 6-13, the cross-network delay of the message is 268.21 msec. This assumes a worst-case solution with one packet released each frame interval, or every 10 msec. The delay is the time necessary to release all but one packet (150 ms) plus the time it takes for the last packet to transit the network (118.2 msec). When the terminals and transmission loops adjacent to the network are considered, the overall delay increases to 25.28 sec.

A satellite link between nodes B and C would cause the propagation delay for that link to become

$$T_{PTR} = \frac{24000 \text{ miles}}{\frac{.6 \text{ kilometers}}{\text{mile}} \times 3 \times 10^8 \text{ m/sec}} = 266.67 \text{ msec.}$$

This would result in a cross-network delay of 530.63 msec for the interactive data call.

If the terminal and computer were capable of 2400 BPS and 64 kBPS, respectively, the overall delay (including loops and terminals) would decrease from 25.28 seconds to 1.132 seconds without the satellite link and 1.395 seconds with the satellite link included. Substituting a 9600 BPS terminal for the 2400 BPS terminal would then cause this overall delay to decreae to 507.79 msec. Even with the satellite link, the delay (707.21 msec) would still be less than a second.

6.5.3 Impact of Transmission Rate on Cross-Network Delay

In order to evaluate the impact of the lower transmission rate used in the experimental system, a comparison of cross-network delays for various packet lengths was made for the 230.4k and 1.544 MBPS transmission links. The following constraints were specified to allow for a meaningful interpretation of the results:

a. One packet is transmitted

b. Class II transmission rates are 20 percent and 50 percent of the carrier transmission rate

c. The bit error rate is 10^{-4}

d. Packet lengths vary from 224 to 1024 bits

e. The transmission of packets in the error environment is described by the negative binomial law. A confidence level of 99.7 percent is used.

f. Maximum packet rate is 100 packets/second

g. The utilization factor is 70 percent.

The results are given in Figure 6-14. This shows the impact on the cross-network delay when a 230.4 kBPS trunk transmission rate is substituted for the T1 carrier rate used in Phase I. For these typical packet lengths, the curves track one another closely. At the packet length of 224 bits (the suggested operating condition), the impact of the lower transmission rate on the delay is minimal (less than 4 percent).

6.5.4 Cross-Network Delay for CCIS Message Sequence

In this example we will examine the cross-network delay of the CCIS message sequence associated with setting up a Class I voice call. The maximum CCIS message length of 232 bits is used and corresponds to a 23.2 kBPS transmission rate. Trunk sending and receiving terminal delays are 10.259 msec and 0.28 msec, respectively. Queue waiting time and cross-office delays are taken to be the same as those specified in Figure 6-12. The call scenario is as follows (see Figure 6-15):

a. Calling subscriber at switch A dials last digit

b. Reserve A-B Channel (F)*

c. Reserve B-C Channel (F)/A-B channel reserved (B)

d. Reserve C-D Channel (F)/B-C channel reserved (B)

*F = forward direction, B = backward direction

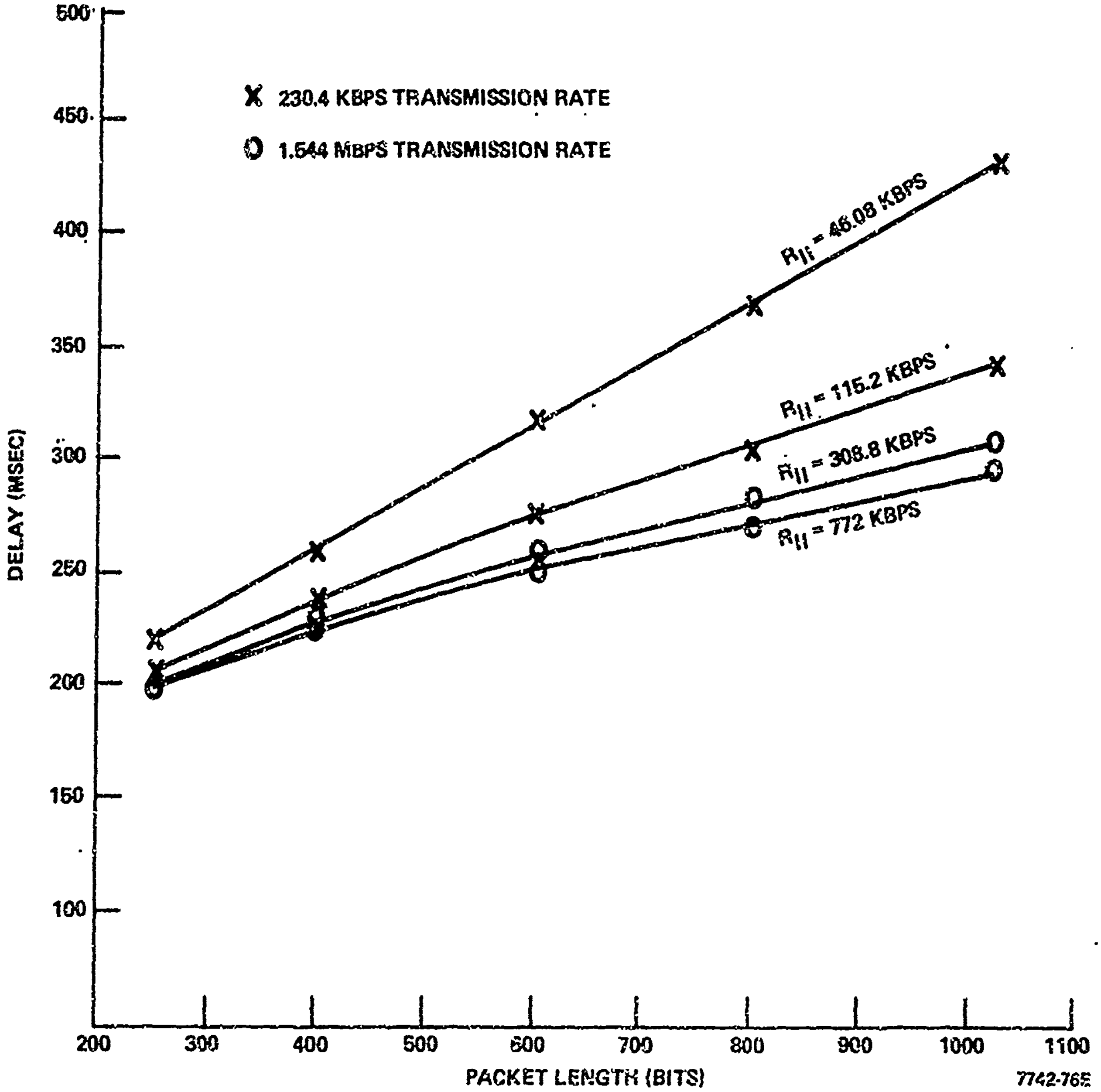


Figure 6-14. Comparison of Cross-Network Delay as a Function of Packet Length

e. Ring forward to subscriber

f. Ringback CCIS message D-C (B)

g. Ringback CCIS message C-B (B)

h. Ringback CCIS message B-A (B)

i. Called subscriber goes off-hook

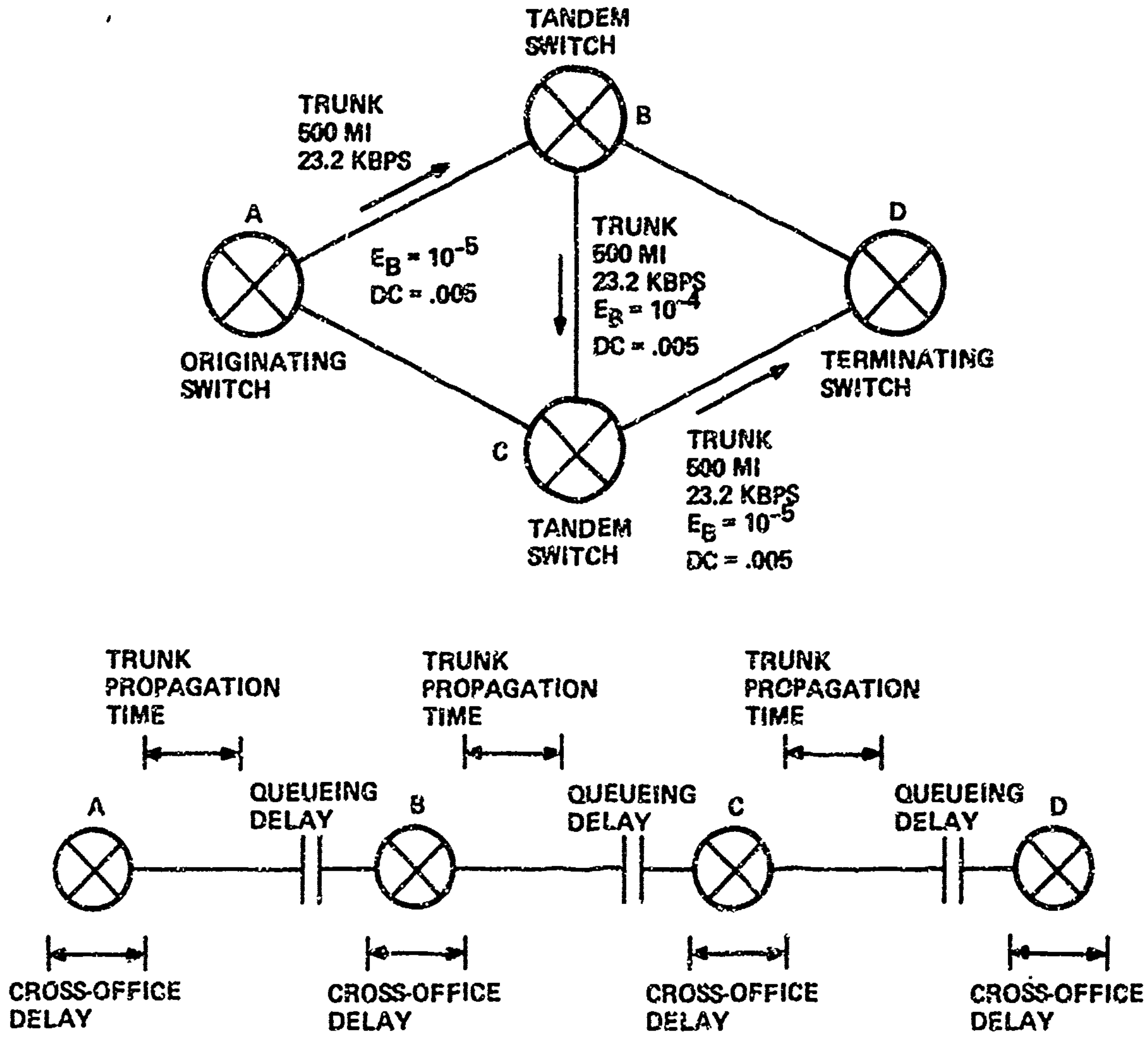
j. Allocate D-C channel (B)

k. Allocate C-B channel (B)/D-C channel allocated (F)

l. Allocate B-A channel (B)/C-B channel allocated (F)

m. A-B channel allocated (F), ringback removed, 2-way conversation enabled.

Seven CCIS messages (packets) are sent in one direction during the call setup. However, to be 99.7 percent confident that 7 messages will be received correctly in this error environment, an average of 12 messages must be transmitted. Thus, 12 single CCIS messages multiplied by the cross-network delay of each message yield an overall delay of 1.608 seconds for call setup time (see Figure 6-15).



	ORIGINATING SWITCH	TRUNK CIRCUIT	QUEUE	TANDEM SWITCH	TRUNK CIRCUIT	QUEUE	TANDEM SWITCH	TRUNK CIRCUIT	QUEUE	TERMINATING SWITCH
TRUNK RECEIVING TIME				0.28			0.28			0.28
PROCESSOR DELAY	21.0			20.0			20.0			21.0
TRUNK SENDING TIME	10.259			10.259			10.259			
TRUNK PROPAGATION TIME		4.25			4.25			4.25		
QUEUEING DELAY			2.54			2.54			2.54	

dimensions in msec.

CROSS-NETWORK DELAY (SINGLE CCIS MESSAGE) = 133.987 MSEC
CROSS-NETWORK DELAY (ENTIRE MESSAGE SEQUENCE) = 1.608 SEC (99.7% CONFIDENCE LEVEL)
CALL SETUP TIME (INCLUDING SUBSCRIBER SIGNALING & RINGING TIME) = 13.608 SEC

7776-76E

Figure 6-15. Cross-Network Delay for CCIS Message Sequence

SECTION 7

COST ESTIMATES FOR EXPERIMENTAL HARDWARE AND OFF-LINE SOFTWARE

7.1 SYSTEM BASIS FOR COSTING

Each nodal system is assumed to contain a nodal overhead equipment group, a link termination equipment group, and an access equipment group. The laboratory network will require one master clock to operate all nodes. Data memory sizes required at link and access ports are indicated in Figure 3-3, where the program memory and peripheral interfaces needed by each of the distributed processors are also shown.

Estimates include three ASR 33 teletypes per node. These teletype machines would function as a Class II data terminal, a Class I/II dummy traffic controller, and a vehicle for node statistical printouts and processor program administration. In the minimum two-node network configuration, only one link termination equipment group is required per node. Therefore it is anticipated that only one full-size equipment rack will be needed. With the addition of a second link termination group per node, as in the three- and four-node network configurations, a second equipment rack would be required at each node.

7.2 EQUIPMENT VENDORS SURVEYED

The information on network costing has been compiled from data derived from Digital Equipment Corporation for the LSI-11 microcomputer, and from Data General Corporation for the μNOVA microcomputer family. Costs have been quantity discounted on the basis of a minimum experimental network of two nodes.

7.2.1 DEC LSI-11

The LSI-11 microcomputer family, manufactured by Digital Equipment Corporation, contains a complete line of hardware, software support, and an off-line software development system. To the hardware cost estimates is added an amount of $13,751.00 for off-line software development. This development system incorporates the BASIC language, has a multi-user capacity for two DECwriter units and a DECprinter, and can remotely transfer a developed program via a serial line interface (SLI) into a distant processor.

7.2.2 Data General μNOVA

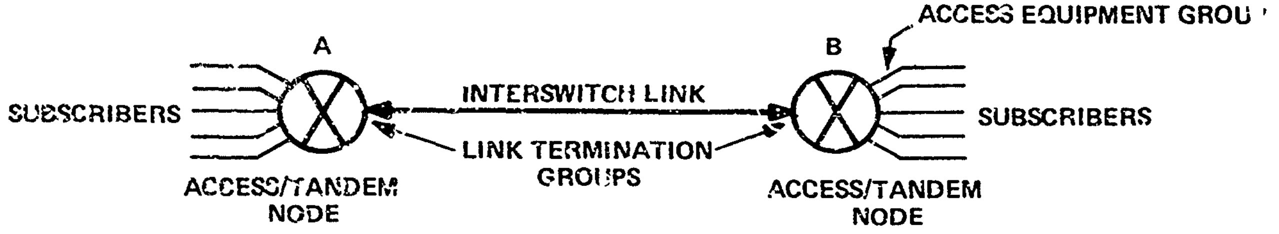
The NOVA microcomputer family, manufactured by Data General Corporation, contains a complete hardware line with some software support. Data General has indicated that an off-line software development system, having a capability similar to the DEC off-line development system, would cost approximately $20,000 to $25,000. Consequently, the network cost estimates include an average cost of $22,500 for Data General off-line software development.

7.3 COST SUMMARY

Cost estimates were made of the experimental SENET-DAX model for various network configurations. These costs, based upon the detailed cost information presented in Appendix II, are summarized in Tables 7-1, 7-2, and 7-3 for two-, three-, and four-node experimental systems. The information presented here pertains only to processor equipment costs and to off-line software development system costs. Note that costs for development of on-line operational software and other non-recurring engineering and development have not been included in this estimate. Neither has the cost of terminal equipment. Preliminary hardware costs of other equipment remaining to be designed in detail (nests, boards, etc.), have also been estimated (see Appendix II). In Appendix II, a listing of components that make up the respective groupings is presented along with itemized costing information. By specifying the equipment needed for a particular network configuration, it is possible to arrive at the cost summaries in Tables 7-1, 7-2 and 7-3 from the cost data supplied in Appendix II.

It appears that both microcomputers are capable of satisfying the operational hardware and software system requirements of the experimental model at a comparable network cost. Further investigation would be necessary with respect to the performance of comparative application-oriented benchmark programs before a final processor choice could be made.

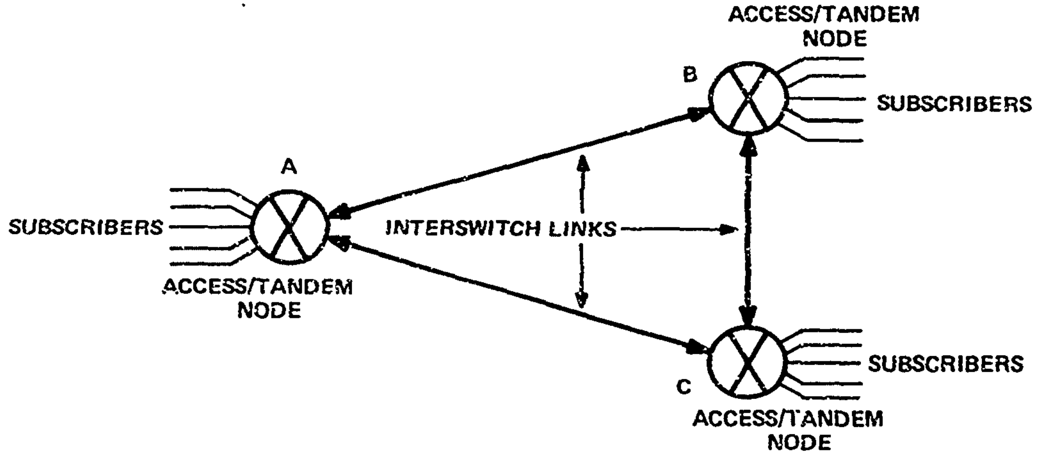
TABLE 7-1. EXPERIMENTAL TWO-NODE NETWORK
COST SUMMARY



SYSTEM EQUIPMENT	NO.	COST	
		LSI-II MICROCOMPUTER TECHNOLOGY	μ NOVA MICROCOMPUTER TECHNOLOGY
ACCESS EQUIPMENT GROUP	2	$31,566	$31,296
NODE POWER SUPPLY & CABINET	2	1,500	1,500
LINK TERMINATION GROUP	2	23,438	21,848
NODAL PROCESSOR GROUP	2	14,168	13,622
DOUBLE OVEN CLOCK	1	800	800
PROM PROGRAMMER	1	1,700	1,700
OFF-LINE SOFTWARE DEVELOPMENT SYSTEM	1	13,751	22,500
TOTAL		$86,923	$93,266

7985-76E

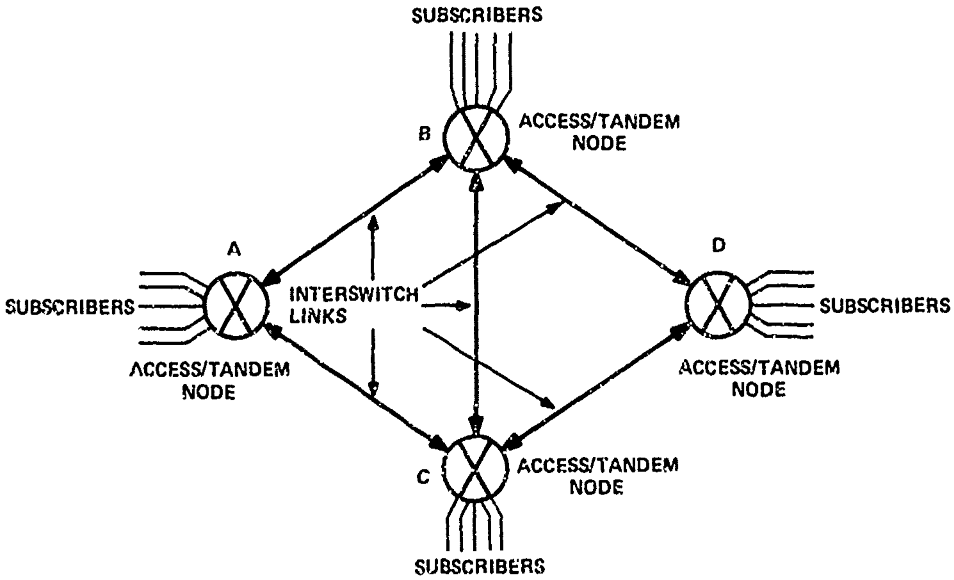
TABLE 7-2. EXPERIMENTAL THREE-NODE NETWORK COST SUMMARY



SYSTEM EQUIPMENT	NO.	COST	
		LSI-11 MICROCOMPUTER TECHNOLOGY	μ NOVA MICROCOMPUTER TECHNOLOGY
ACCESS EQUIPMENT GROUP	3	$47,349	$46,944
NODE POWER SUPPLY & CABINETS	3	3,900	3,900
LINK TERMINATION GROUP	6	70,314	65,544
NODAL PROCESSOR GROUP	3	21,252	20,433
DOUBLE OVEN CLOCK	1	800	800
PROM PROGRAMMER	1	1,700	1,700
OFF-LINE SOFTWARE DEVELOPMENT SYSTEM	1	13,751	22,500
TOTAL		$159,066	$161,821

7966-76E

TABLE 7-3. EXPERIMENTAL FOUR-NODE NETWORK COST SUMMARY



SYSTEM EQUIPMENT	NO.	COST	
		LSI-II MICROCOMPUTER TECHNOLOGY	μ NOVA MICROCOMPUTER TECHNOLOGY
ACCESS EQUIPMENT GROUP	4	$ 63,132	$ 62,592
NODE POWER SUPPLY & CABINETS	4	5,200	5,200
LINK TERMINATION GROUP	10	117,190	109,240
NODAL PROCESSOR GROUP	4	28,336	27,244
DOUBLE OVEN CLOCK	1	800	800
PROM PROGRAMMER	1	1,700	1,700
OFF-LINE SOFTWARE DEVELOPMENT SYSTEM	1	13,751	22,500
TOTAL		$230,109	$229,276

7987-76E

SECTION 8

FURTHER NETWORK CONSIDERATIONS

8.1 GENERAL

The next logical step in this program is the experimental implementation of the SENET-DAX concept of a limited multinode network. This would allow the techniques established during the phase just concluded and during the previous Phase I study to be implemented and evaluated in software and hardware, and analytic tradeoffs investigated experimentally. Certain additional areas, however, have great potential application toward the realization of an experimental SENET-DAX network and are discussed in the following sections. These include communication over satellite links and interface with the ARPA network.

8.2 SATELLITE COMMUNICATIONS

The networks of interest for the application of the SENET-DAX concept are already of global proportions and it appears that this trend will continue in future communication networks. For example, both the AUTOVON and AUTODIN communication networks are currently of intercontinental scope, while networks such as the ARPANET are already of transcontinental scope. In order to provide such global networks economically, extensive use of satellite links to supplement or replace landline or radio links between switching nodes is foreseen [5]. Over the past few years satellite communication has experienced rapid advances resulting in a continuing decrease in satellite cost.

In fact, studies have shown that a potential cost savings of perhaps as much as 50 percent are achievable from the utilization of satellite capacity at the expense of terrestrial capacity [17]. Terrestrial facilities would still be necessary in order to achieve a high level of survivability for critical functions such as command and control. Thus it would still be necessary to provide switching facilities in the access area to switch between terrestrial transmission systems. However, the proportion of traffic carried by terrestrial circuits relative to that carried by the satellite would decrease.

Satellite systems organized in a multiple access arrangement in which multiple users communicate over common satellite channels are expected to be very attractive for use in packet data communication networks [23]. These can be divided into two general categories. In the first category bandwidth is divided among users by FDM or TDM techniques into fixed allocations with each user permanently assigned a fixed portion of the total available transponder bandwidth. In the second category user channel assignments are dynamically allocated, in response to user requests, either by means of a central control station, the ground stations themselves, or a random access method of assignment. The class of dynamic channel assignment which does not use a central control is the one which has generated considerable interest for packet data communications. It should be noted however, that further study, particularly in the area of system control during crisis conditions, is necessary before the use of satellite ground stations in military networks can be made widespread.

The combining of integrated voice/data switching with satellite capabiity could have far-reaching consequences on the network topology and architecture. Figure 8-1 illustrates the evolutionary possibilities for our experimental four-node network in which a fully-connected distributed satellite network is overlayed onto the present network structure. The SENET-DAX nodes typify small size access switches feeding a highly interconnected backbone structure of tandem switches and interconnected trunk circuits. The latter could be AN/TTC-39's, AUTOVON switches, or other elements of the present DCS network. The configuration allows the extension of integrated voice and data switching and multiplex techniques into the access area and pro ides satellite communication capability as well. It also provides an experimental tool for extending the AUTODIN II network to have overseas packet capabilities and for providing medium-sized access area concentrators for later extensions of the Phase II Secure Voice program [5]. Eventually a network composed of satellite links and access switches would emerge in which subscribers in the access area would have significantly better user service capability. The backbone structure, however, could continue for an indefinite transition period and evolve as future requirements change.

During the experimental phase one satellite link at a T2 carrier rate (6.312 MBPS) will provide the necessary capacity for full connectivity of a four-node integrated voice/data network.* A dynamic

---

* Full connectivity of a four-node network requires six links, each operating at a transmission and receiving rate of 230.4kBPS. Therefore, a capability of 2 x 230.4 x 6 = 2.7648MBPS, is required.

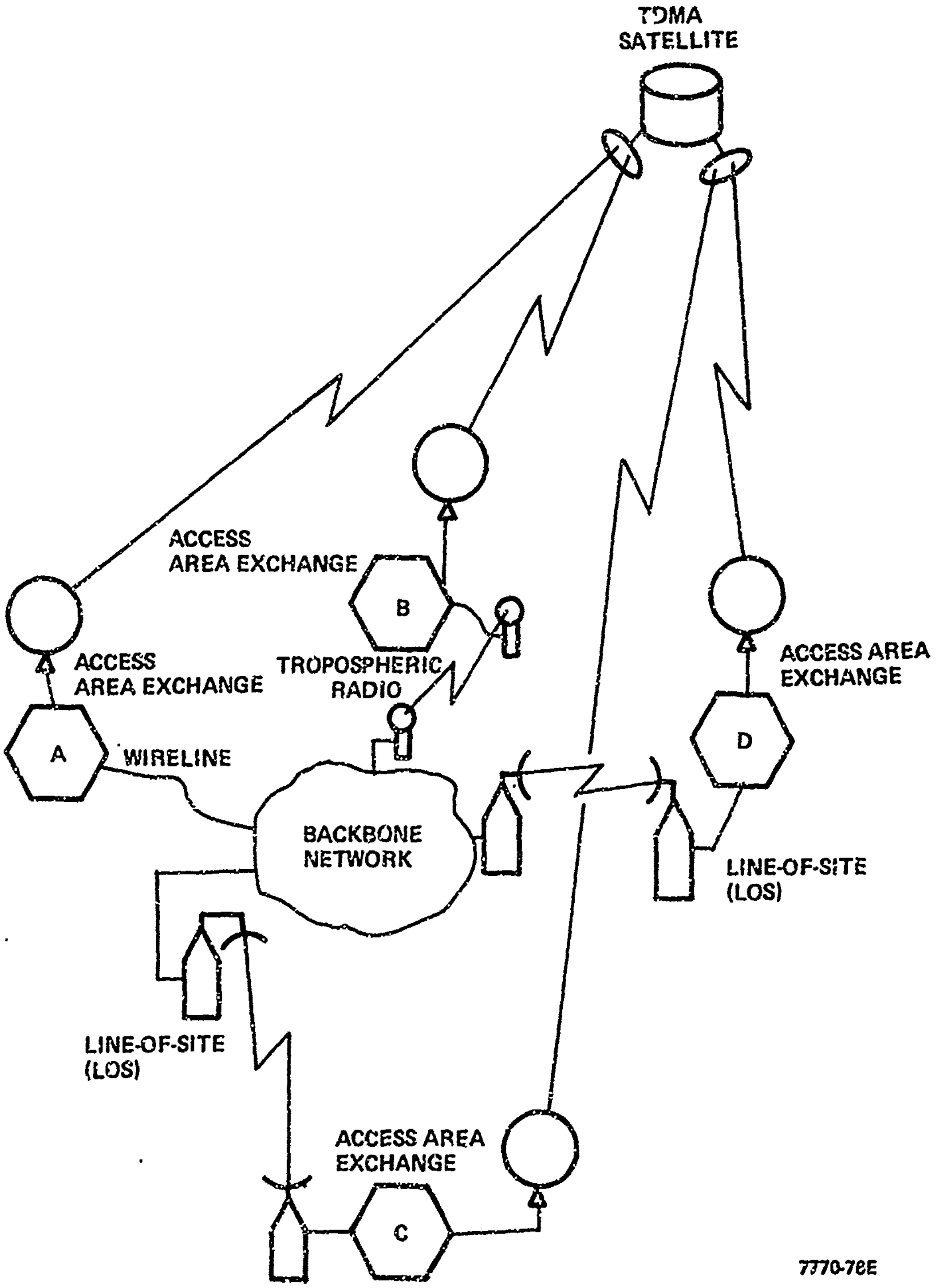


Figure 8-1. Experimental Network Structure with Satellite Capability

variable transmission rate, software controllable, from 230.4kBPS down to some minimum, could also be developed so that only the capacity which contains real information is transmitted over the satellite link.

8.3 INTERACTION WITH THE ARPA COMPUTER NETWORK

A potential area of application during an experimental phase is the interaction between the four-node experimental network and the ARPA network. The ARPA network is a complex, distributed, packet-switching network implemented by a set of small processors called Interface Message Processors (IMP's) interconnected by 50 kBPS common carrier circuits [6,10,16]. Its primary goal is to permit personnel and program at any node in the network to access data and software from remote processors elsewhere in the network. One possible experiment which could be used to demonstrate the flexibility of the SENET-DAX approach would be to carry packet data traffic generated by the ARPA network over SENET-DAX transmission links along with voice and data traffic generated by the SENET-DAX system. This situation is depicted in Figure 8-2. A pair of widely separated but directly connected nodes in the ARPA network are chosen and outgoing traffic on the link connecting these nodes is tapped on the trunk side. This traffic, which represents actual data being transacted in the ARPA network, transits the SENET-DAX experimental network and is delivered to a tap on the trunk side of the incoming ARPA node where it is to be decoded. This would demonstrate that ARPA network traffic can be transmitted as a subset of the voice and data traffic transmitted by the SENET-DAX network. ARPANET data traffic could be

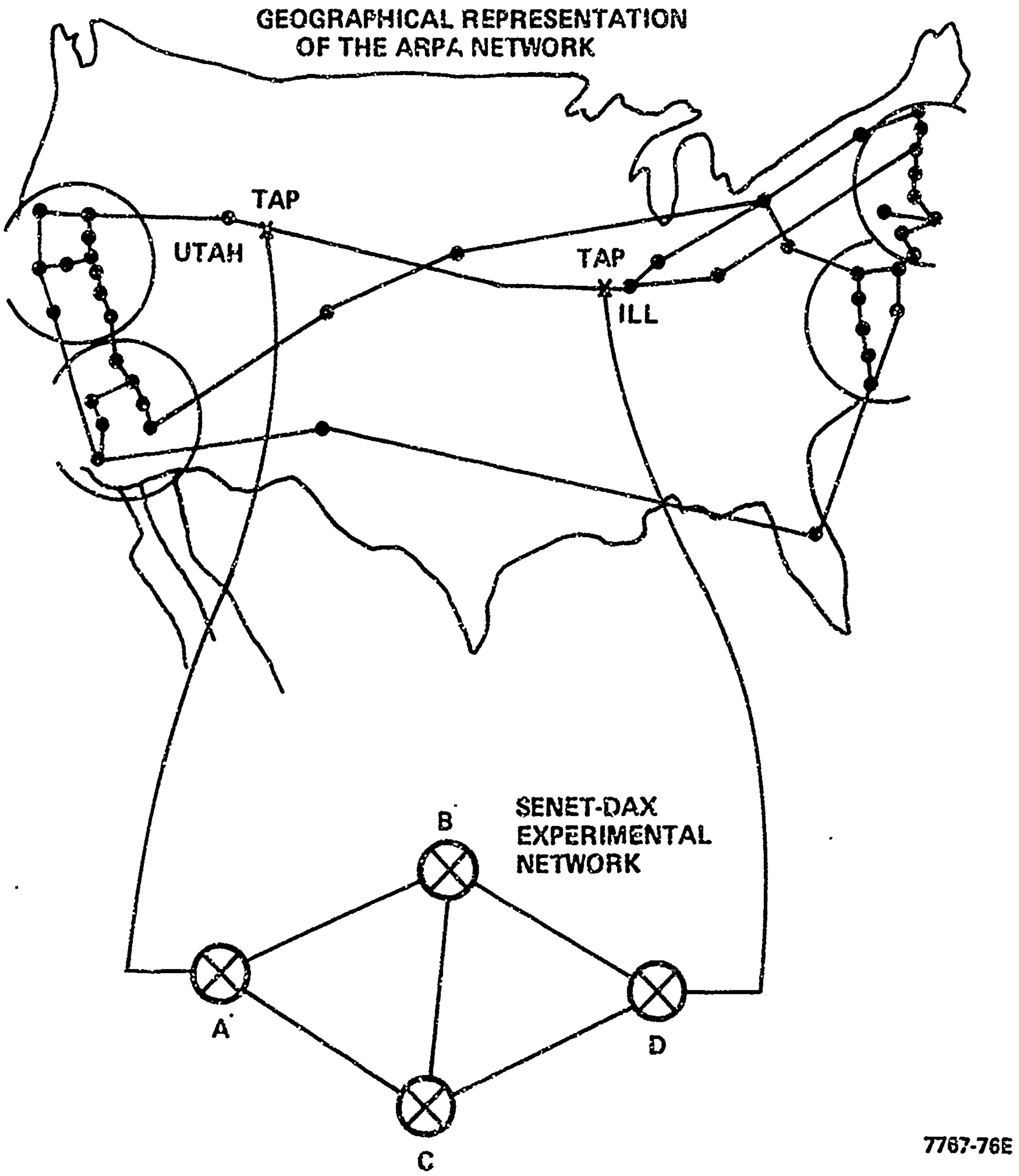

GEOGRAPHICAL REPRESENTATION OF THE ARPA NETWORK
TAP
UTAH
TAP
ILL
SENET-DAX EXPERIMENTAL NETWORK
B
A
D
C
7767-76E

Figure 8-2. Interaction of the SENET-DAX Experimental Network with the ARPA Network

transmitted either as a 50 kBPS "super-packet in the Class II region or else a full period connection in the Class I region could be reserved for ARPANET purposes.* At the receive side data would be fed to the terminating IMP and decoded in normal fashion. An alternate method is for the ARPA node to receive traffic as it normally does while a hard copy is generated for comparison with the traffic sent over the SENET-DAX network. A comparator which could account for the different delays in the two paths could verify that the same information was received intact over the separate routes.

In this experiment SENET-DAX nodes need not be concerned with ARPANET protocol, format, error control, or routing strategy. The entire SENET-DAX network appears as just another transmission media to the ARPA network. This would not be the case if the interface were on the terminal side of the ARPA node. Here we would be required to emulate the IMP, i.e., packetize, interface with the host terminal and host computers, detect packets, etc. Since the ARPA network already accomplishes this function, no advantage would be gained from doing this.

It may be possible to choose nodes which are co-located with satellite ground stations. Traffic could then be tapped from the ARPA network, combined with voice and data traffic from the SENET-DAX experimental network, and fed into the ground station for transmission

* Although the transmission rate in the ARPA network is 50 kBPS, actual data traffic is on the order of 20 kBPS. Therefore, it may be possible to reserve less capacity to handle ARPANET traffic or even to make this allocation dynamic.

over the satellite link. At the receive side the information would be delivered to a SENET-DAX node which would break out the ARPANET portion and input it to the receive node to be decoded.

SECTION 9

REFERENCES

1. Bobeck, A. H., and Scovil, H. "Magnetic Bubbles", Scientific American, June, 1971.

2. Chu, W. "Optimal Message Block Size for Computer Communications with Error Detection and Retransmission Strategies:, IEEE Transactions on Communications, October, 1974.

3. Coviello, G., and Vena, P. "Concept for an Integrated Circuit-and-Packet-Switched Telecommunications System",DCA Technical Comment No. 8-75, January, 1975.

4. Coviello, G. and Vena, P. "Integration of Circuit/Packet Switching by a SENET (Slotted Envelope Network) Concept", NTC, New Orleans, LA, 1975.

5. Coviello, G. "Unifying System Engineering", ICC Convention, Philadelphia, PA, June 1976.

6. Heart, F. E., et al. "The Interface Message Processor for the ARPA Computer Network", Spring Joint Computer Conference, 1970.

7. Heart, F. E., et al. "A new Minicomputer/Multiprocessor for the ARPA Network", National Computer Conference, 1973.

8. Hiles, J. "A Review of Selected Sessions, COMPCON 76 SPRING - Session 2, IEEE Journal of Computers, April 1976.

9. Fischer, M. and Harris, T. "Analysis of an Integrated Circuit and Packet Switched Telecommunications System", DCEC Technical Note 6-75, January 1975.

10. Kahn, R. E. "Resource-Sharing Computer Commuications Networks", Proceedings of the IEEE, November 1972.

11. Kummerle, K. "Multiplexor Performance for Integrated Line and Packet Switched Traffic", ICCC 1974 Conference, Stockholm, Sweden.

12. Leach, G. C. "State of the Art in Microprocessor Software", Electro 76, Boston, Mass, May 1976.

13. Martin, J. Telecommunications and the Computer, Prentice-Hall, 1976.

14. Ogdin, J. L. "Microcomputers: Promises and Practices, Revisited", Electro 76, Boston, Mass, May 1976.

15. Re[illegible], S. L. "Cost, Performance and Size Trade-offs for Different Levels in a Memory Hierarchy", IEEE Journal of Computers, April 1976.

16. Roberts, L. G. and Wessler, B. D. "Computer Network Development to Achieve Resource Sharing", Spring Joint Computer Conference, 1970

17. Rosner, R. "Optimization of the Number of Ground Stations in a Domestic Satellite System", EASCON, Washington, D. C., September 1975.

18. Schmitz, H., Saxton, T. and Huang, C. "A Real-Time Associative Processor for Integrated Voice/Data Compression and Multiplexing", NBS-ACM Annual Technique Symposium, 1976.

19. Speliotis, D. "Bridging the Memory Access Gap", NCC AFIPS Conference Proceedings, Vol. 44, ' y 1975.

20. Spain, R. and Marino, M. "Magnetic Film Domain - Wall Motion Devices", IEEE Transactions on Magnetics, September, 1970.

21. Syski, R. Congestion Theory in Telephone Systems, Oliver and Boyd, London, 1960.

22. Wulf, W. A. and Bell, C. "Cmmp - A Multi-Mini Processor", Proc. AFIPS Conference - Fall Joint Computer Conference Paper - 1972, Vol. 41.

23. Yuill, S. "A Survey of Multiple Access Satellite Packet Communication", NBS-ACM Annual Technique Symposium, 1976.

24. Advanced Data Communication Control Procedures, (ADCCP), Proposed American National Standard, Fifth Draft, April 1976.

25. The Application of Multiple Processor Computer Systems to Digital Communications Networks, Carnegie-Mellon University, DCA Contract DCA100-75-C-0066, June 1976.

26. SENET-DAX Study: Final Report, DCA Contract No. DCA-100-75-C-0071. GTE Sylvania, Needham, Mass. June 1976.

27. Standards for Long Haul Communications; Switching Planning Standards for the Defense Communications System. Defense Communications Agency, Proposed MIL-STD-187-310, March 1976.

APPENDIX I

TIMING ANALYSIS TABLES

TABLE I-1. ASSUMPTIONS FOR TIMING ANALYSIS

Traffic

- 500 Packets/Sec (absolute max)
- 200 Packets/Sec (average busy hour)
- 13 Calls/100 Sec [over 3 links]
  Total Calls
- 6.5 local calls/100 sec

From
Performance
Analysis
Section 6

Interrupts

- Real-time Clock interrupt of 1 ms
- 4 ms interrupt on Control Scheduler

Instruction Execution Time/Interrupt Service

- 3 μsec/average instruction execution time
- 10 μsec Interrupt Service Time

Capability of Micro-Processor

- Simple 1 word accumulator instructions

  Load, Store, Bit test, Set bit, Add/Subtract

  Branch conditional, unconditional (on contents)
- Subroutine
- Indexing
- No multi-indirect instruction capability - 1 level only

TABLE I-2. CALCULATIONS FOR TIMING ANALYSIS FOR LINK INPUT PROCESSOR

Real-Time Interrupt Routine

(50 instr/int) (3 µsec) + 10 µsec/int (1000 int/sec)

= 160 ms/sec or 167 percent loading

Control Scheduler Module

(100 instr/cycle) (3 µsec/instr) + 10µ/int) (250 int/sec)

= 77.5 ms/sec or 7.75 percent loading

Free Memory Background Scan-List Management

OVERHEAD LOOP FOR EACH ADDRESS

(10 instr/overhead) (3 µsec)
+ [(30 instr/addr) (3 µ/instr) (250 address)]
= 22.5 ms/sec = 2.25 percent loading

Packet Analysis Background Scan

Instr Exec Routine	time for	# of packets/sec
(200 instr/exec)	(3 µsec/int)	(500 packets/sec)

= 300 ms/sec = 30 percent loading

Translation Subroutine

Instr/Packet	x	Instr Time	x	# of packets
(30 instr/packet)		(3 µsec/instr)		(500 packet/sec)

= 45 ms/sec = 4.5 percent loading

Routing Subroutine

Instr/Packet	x	Instr Time	x	# of packets
(28 instr/packet)		(3 µsec/instr)		(500 packets/sec)

= 42 ms/sec = 4.2 percent loading

CCIS Packet Generation

Instr/Packet x Instr Time x # of CCIS packets/sec

(80 instr/packet) (3 μsec/instr) (13 calls/100sec x 10 messages)

= .312 ms = .03 percent loading (negligible)

Transmit Packet Control

# of instr/packet x Instr Time x # of packets/sec

(40 instr/packet) (3 μsec/instr) (500 packets/sec)

= 60 ms/sec = 6 percent loading

Acknowledge/No Acknowledge

# of Instr/Packet x Instr Time x # of packets/sec

(50 instr/packet) (3 μsec/instr) (500 packets/sec)

= 75 ms/sec = 7.5 percent loading

Maintenance and Statistics

# of instr exec/pass x Instruction Time

(2000 instr/pass) (3 μsec/instr)

= 6.0 ms/sec = .6 percent loading (negligible)

TABLE I-3. CALCULATIONS FOR TIMING ANALYSIS FOR LINK OUTPUT PROCESSOR

Real-Time Interrupt Routine

(50 instr/int) (3 µsec) + 10 µsec/int) (1000 int/sec)

= 160 ms/sec or 16 percent loading

Control Scheduler Module

(100 instr/cycle) (3 µsec/instr) + 10µ/int) (250 int/sec))

= 77.5 ms/sec or 7.75 percent loading

Free Memory Background Scan - List Management

OVERHEAD	LOOP FOR EACH ADDRESS
(10 instr/overh) (3µ/instr) +	[(30 instr/addr) (3µ/instr) (250addr)]

= 22.5 ms/sec = 2.25 percent loading

Background Queue Entry

# of instr	x	Time for exec	x	# of packets
(60 instr/packet)		(3µsec/instr)		(500 packets)

= 90 ms = 9.0 percent loading

Timeout Processing

# of instr	x	Time for exec	x	# of bad packets/sec
(50 instr/packet)		(3 µsec/instr)		(1.1 packets/sec)

= 152 µsec = 1.5 percent loading

Output Address Generator

# of instr/packet	x	Instruction time	x	# of packets/sec
(40 instr/packet)		(3 µsec/instr)		(500 packets/sec)

= 60 ms = 6 percent loading

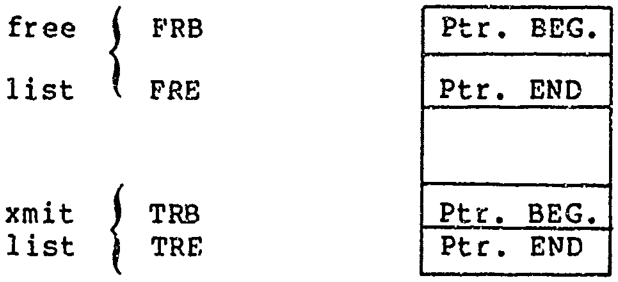
Maintenance & Statistics

# of instr exec/pass	x	Instruction time
(2000 instr/pass)		(3 µsec/instr)

= 6.0 ms/sec = .6 percent loading (negligible)

TABLE I-4. GENERALIZED CODE SET SAMPLE OF LIST PROCESSING/TIMING ALGORITHM

Problem: Remove ith element from one chained list and add the element to the end of another list. (e.g. transmit-to-free list exchange)



<u>Generalized Code</u>

Fetch - source, dest.

Stor - source, dest.

( ) - one level indirect addressing

Avg instr exec time 3.5μsec

<u>Assumed Calling Sequence</u>

```
          JSR               R1, REMI
          FRB    or         TRB
          # in list (i)                        (i =∑1 → ∞)
          FRE    or         TRE
```

<u>Subroutine</u>

```
REMI:     STOR              (R1), TMPI         /FRB or TRB  TMP1
          INC               R1
          FETCH             (TMP1), RØ         /A1 in RØ
          STOR              (41), PTR          /i → PTR
          INC               R1
          DEC               PTR                /L → i-1
          BNE               L1 +1              /Branch if not 1st el.
          STOR              RØ, TMP3
          FETCH             TMP1, RØ           /FRB or TRB → RØ
          STOR              RØ, S1             /FRB or TRB → S1
          FETCH             TMP3, RØ           /A1 → RØ
          STOR              RØ, S3             /A1 → S3
          JSR               R1, REMF           /GO TO REMOVE FIRST
S1:       WORD              Ø
          STOR              (R1), S2           /FRE OR TRE → S2
          INC               R1
          JSR               R1, ADDE        /GO TO ADD EL TO END
S2:       WORD              Ø
S3:       WORD              Ø
          RTS
L1:       DEC               PTR
          BEQ               L2             /BRANCH if found ith element
          FETCH             (RØ), RØ       /A2 →RØ
          Br                L1
L2:       STOR              (RØ), FP       /ptr to next element → FP
          STOR              RØ, S3         /add of el. to be
                                             deleted → S3
          TST               FP             /if=0:El. found is last El.
          BNE        L3
          STOR       (R1), S4        /Set up last el. delete
                                     /routine - FRE or TRE  S4
          JSR        R1, REML
S4:       WORD       Ø               /FRE or TRE
          BR         S1 + 1
L3                                   /ith element is neither 1st
                                     nor last
          ADD        BS1Z, RØ        /loc. of back ptr of El to
                                     be deleted in RØ
          STOR       (RØ), TMP3      /Prev El. ptr in TMP3
          FETCH      FP, RØ          /Front pointer of element to
                                     /be deleted
          STOR       RØ, (TMP3)      /adjust front ptr of prev.
                                       El
          FETCH      FP, RØ          /front ptr to next El  RØ
          ADD        BS1Z, RØ        /front of back ptr of next
                                     El in RØ
          FETCH      TMP3, (RØ)      /adjust back ptr of next El.
          BF         S1 + 1
FP:       WORD       Ø
TMP3:     WORD       Ø
```

Timing Algorithm

Based on position of the element within the table (ith position), the following chart provides several timing algorithms as a function of i.

	# OF INSTRUCTIONS	EXECUTION TIME (μsec)
i = 1	45	157.5 μsec
i = last in list	37 + 3i	129.5 + (10.5)i
i = not first or last	33 + 3i	115.5 + (10.5)i

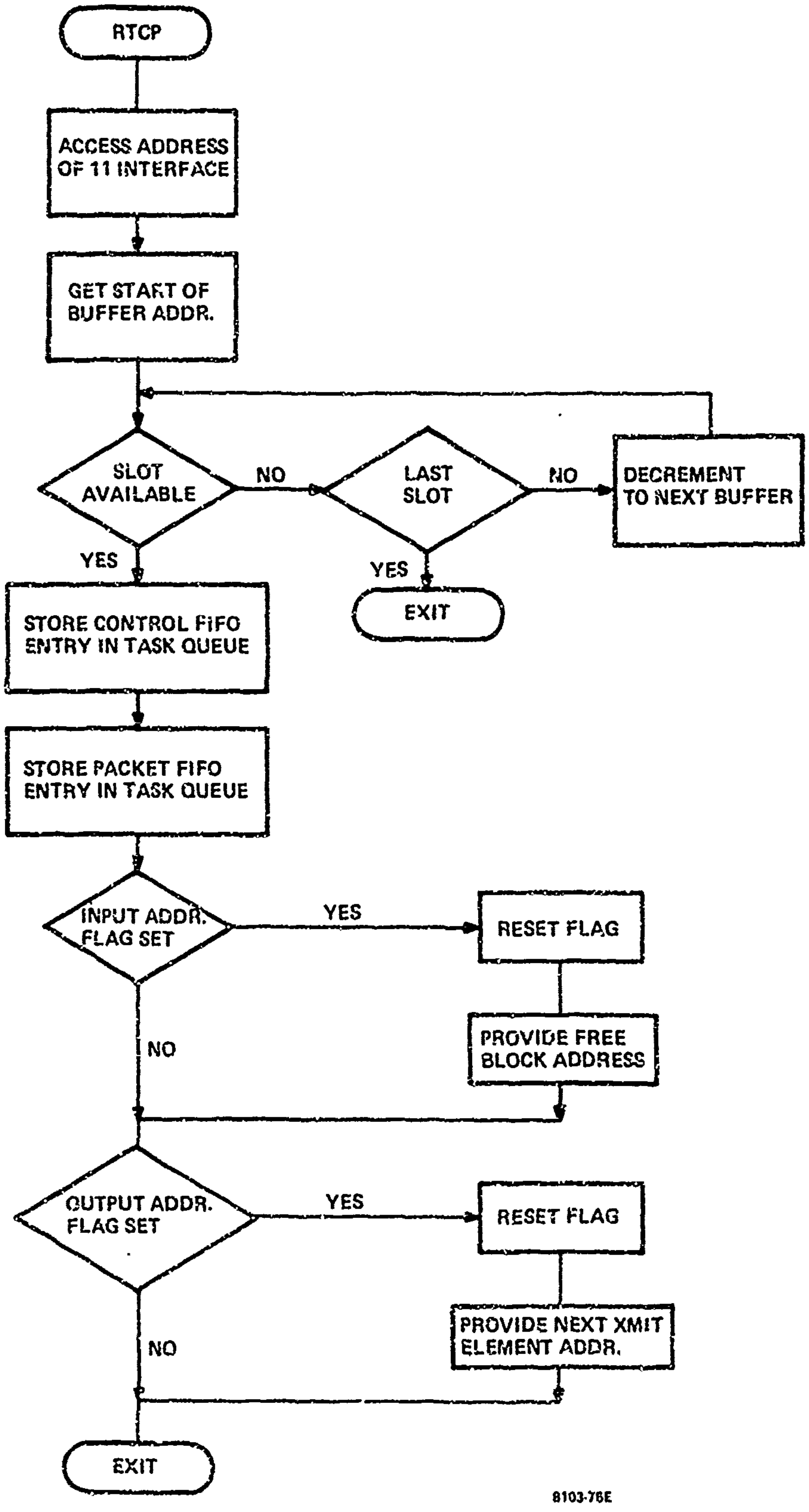

RTCP
ACCESS ADDRESS OF 11 INTERFACE
GET START OF BUFFER ADDR.
SLOT AVAILABLE
NO
LAST SLOT
NO
DECREMENT TO NEXT BUFFER
YES
YES
EXIT
STORE CONTROL FIFO ENTRY IN TASK QUEUE
STORE PACKET FIFO ENTRY IN TASK QUEUE
INPUT ADDR. FLAG SET
YES
RESET FLAG
NO
PROVIDE FREE BLOCK ADDRESS
OUTPUT ADDR. FLAG SET
YES
RESET FLAG
NO
PROVIDE NEXT XMIT ELEMENT ADDR.
EXIT
8103-76E

Figure I-1. Real-Time Clock Interrupt

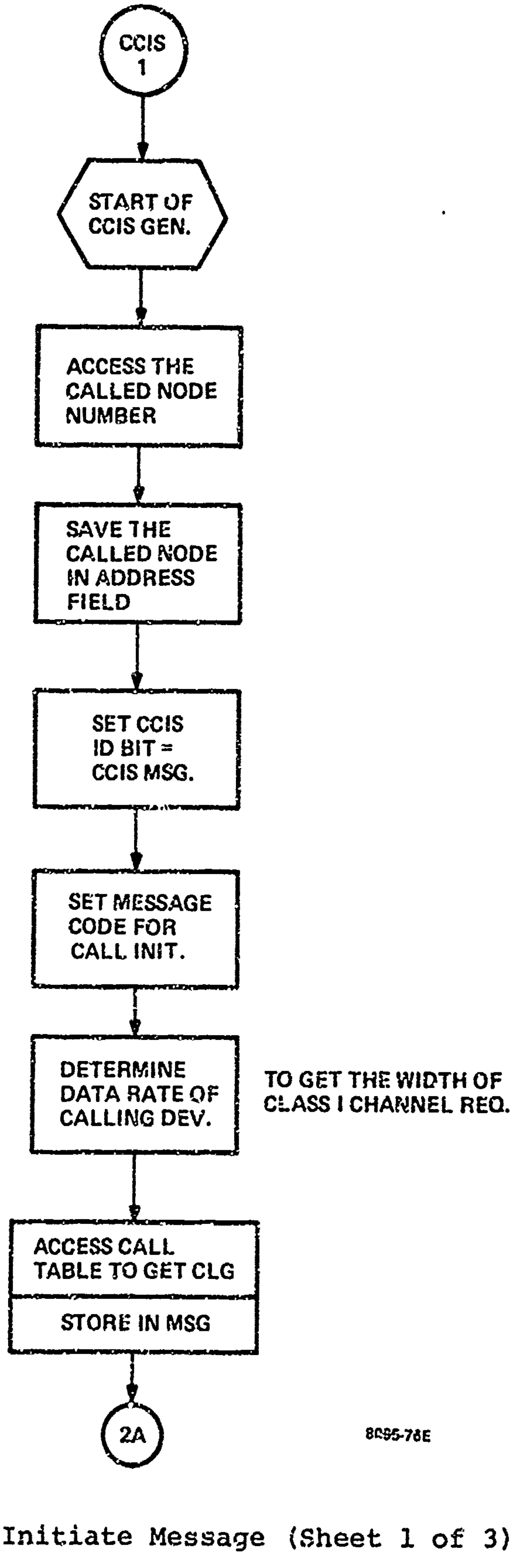

CCIS 1
START OF CCIS GEN.
ACCESS THE CALLED NODE NUMBER
SAVE THE CALLED NODE IN ADDRESS FIELD
SET CCIS ID BIT = CCIS MSG.
SET MESSAGE CODE FOR CALL INIT.
DETERMINE DATA RATE OF CALLING DEV.
TO GET THE WIDTH OF CLASS I CHANNEL REQ.
ACCESS CALL TABLE TO GET CLG
STORE IN MSG
2A
8095-78E

Figure I-2. Call Initiate Message (Sheet 1 of 3)

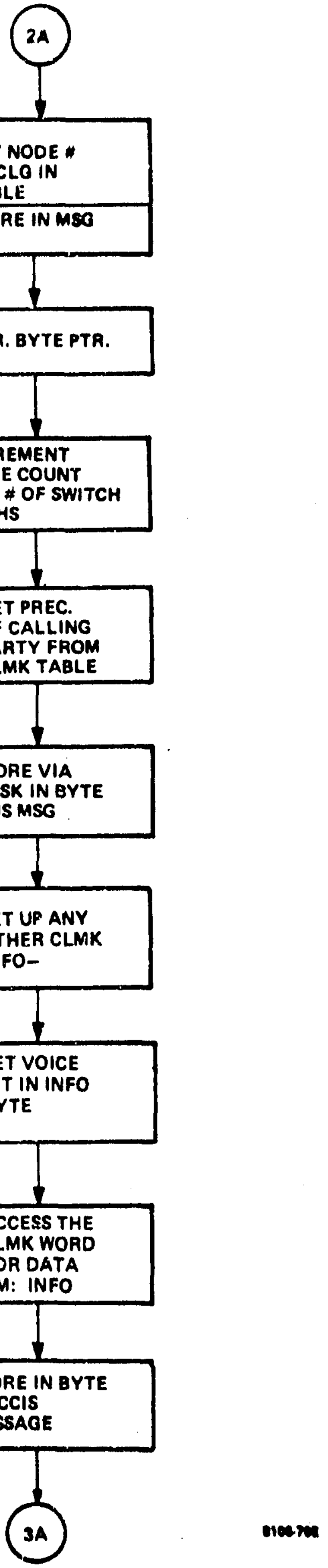


Figure I-2. Call Initiate Message (Sheet 2 of 3)

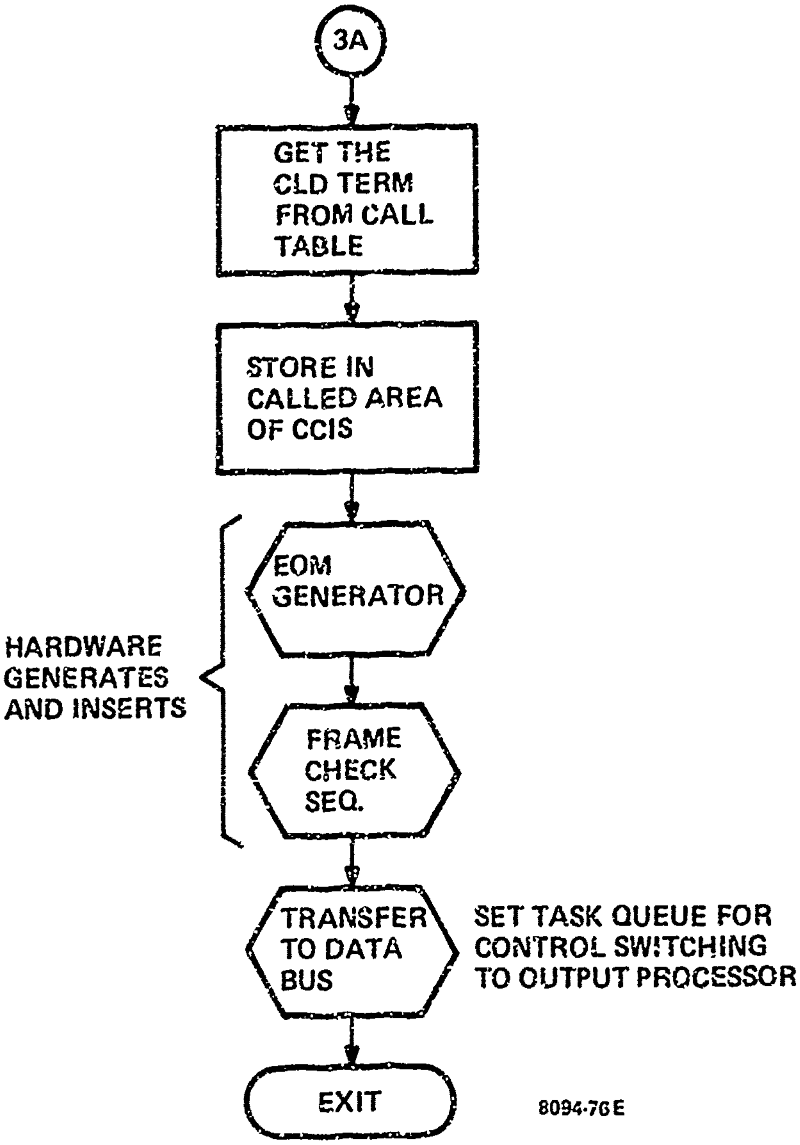

3A
GET THE CLD TERM FROM CALL TABLE
STORE IN CALLED AREA OF CCIS
EOM GENERATOR
HARDWARE GENERATES AND INSERTS
FRAME CHECK SEQ.
TRANSFER TO DATA BUS
SET TASK QUEUE FOR CONTROL SWITCHING TO OUTPUT PROCESSOR
EXIT
8094-76E

Figure I-2. Call Initiate Message (Sheet 3 of 3)

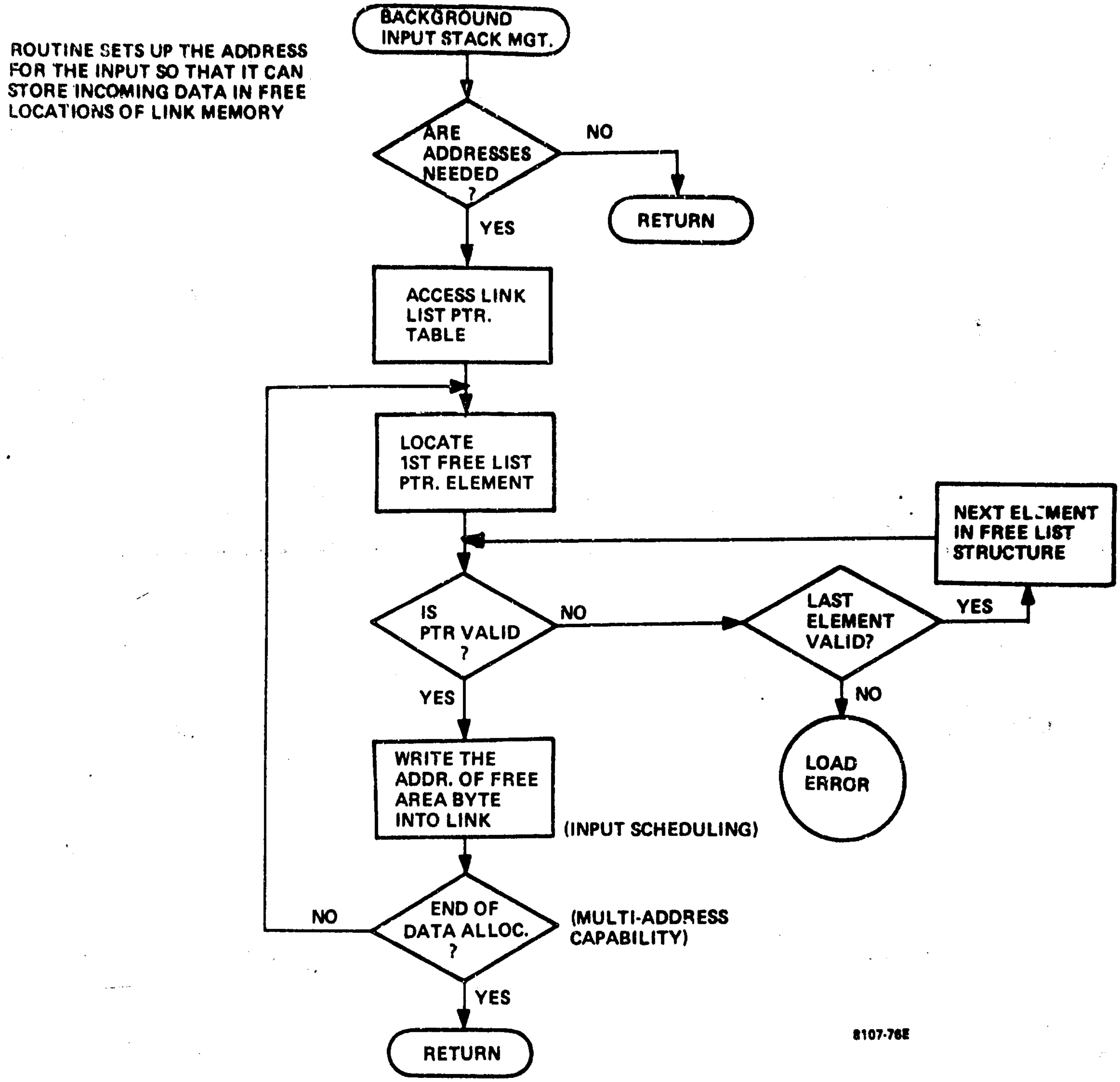

ROUTINE SETS UP THE ADDRESS FOR THE INPUT SO THAT IT CAN STORE INCOMING DATA IN FREE LOCATIONS OF LINK MEMORY
BACKGROUND INPUT STACK MGT.
ARE ADDRESSES NEEDED ?
NO
RETURN
YES
ACCESS LINK LIST PTR. TABLE
LOCATE 1ST FREE LIST PTR. ELEMENT
NEXT ELEMENT IN FREE LIST STRUCTURE
IS PTR VALID ?
NO
LAST ELEMENT VALID?
YES
NO
LOAD ERROR
YES
WRITE THE ADDR. OF FREE AREA BYTE INTO LINK
(INPUT SCHEDULING)
NO
END OF DATA ALLOC. ?
(MULTI-ADDRESS CAPABILITY)
YES
RETURN
8107-76E

Figure I-3. Input Processor - Stack Management

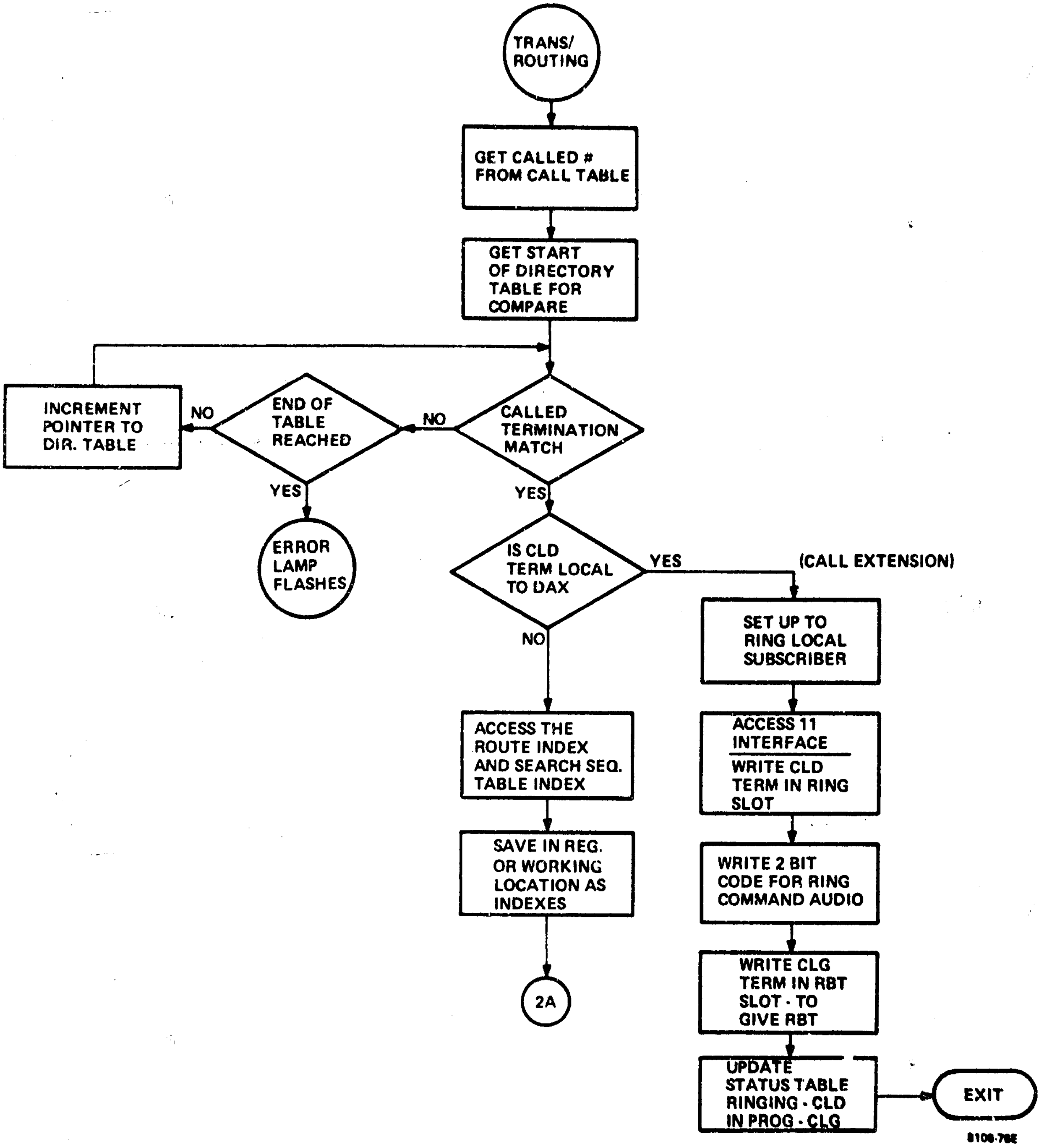

TRANS/ ROUTING
GET CALLED # FROM CALL TABLE
GET START OF DIRECTORY TABLE FOR COMPARE
CALLED TERMINATION MATCH
NO
END OF TABLE REACHED
NO
INCREMENT POINTER TO DIR. TABLE
YES
ERROR LAMP FLASHES
YES
IS CLD TERM LOCAL TO DAX
YES
(CALL EXTENSION)
SET UP TO RING LOCAL SUBSCRIBER
ACCESS 11 INTERFACE
WRITE CLD TERM IN RING SLOT
WRITE 2 BIT CODE FOR RING COMMAND AUDIO
WRITE CLG TERM IN RBT SLOT - TO GIVE RBT
UPDATE STATUS TABLE RINGING - CLD IN PROG - CLG
EXIT
NO
ACCESS THE ROUTE INDEX AND SEARCH SEQ. TABLE INDEX
SAVE IN REG. OR WORKING LOCATION AS INDEXES
2A
8100-78E

Figure I-4. Translation and Routing (Sheet 1 of 3)

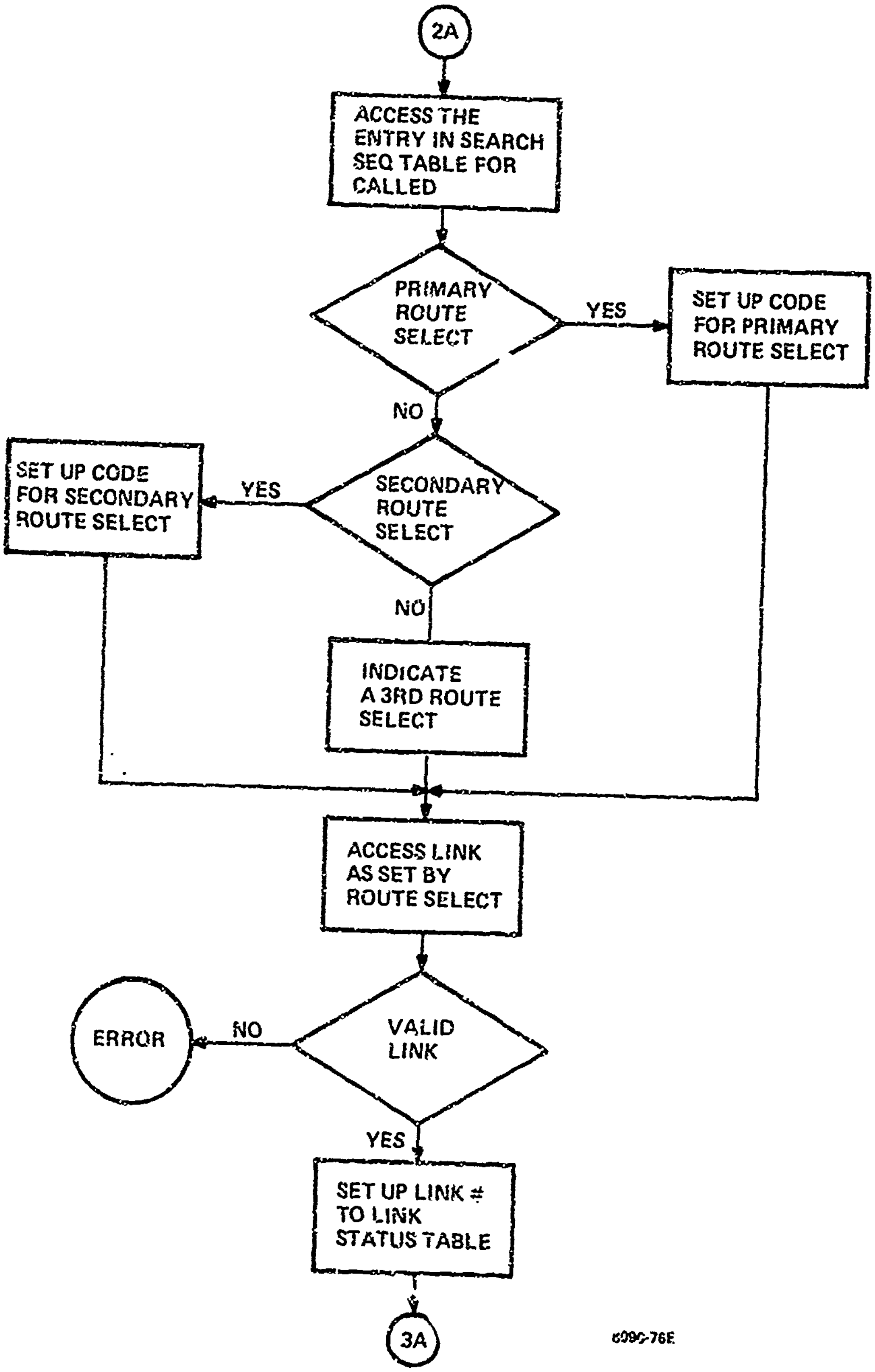

2A
ACCESS THE ENTRY IN SEARCH SEQ TABLE FOR CALLED
PRIMARY ROUTE SELECT
YES
SET UP CODE FOR PRIMARY ROUTE SELECT
NO
SECONDARY ROUTE SELECT
YES
SET UP CODE FOR SECONDARY ROUTE SELECT
NO
INDICATE A 3RD ROUTE SELECT
ACCESS LINK AS SET BY ROUTE SELECT
VALID LINK
NO
ERROR
YES
SET UP LINK # TO LINK STATUS TABLE
3A
6090-76E

Figure I-4. Translation and Routing (Sheet 2 of 3)

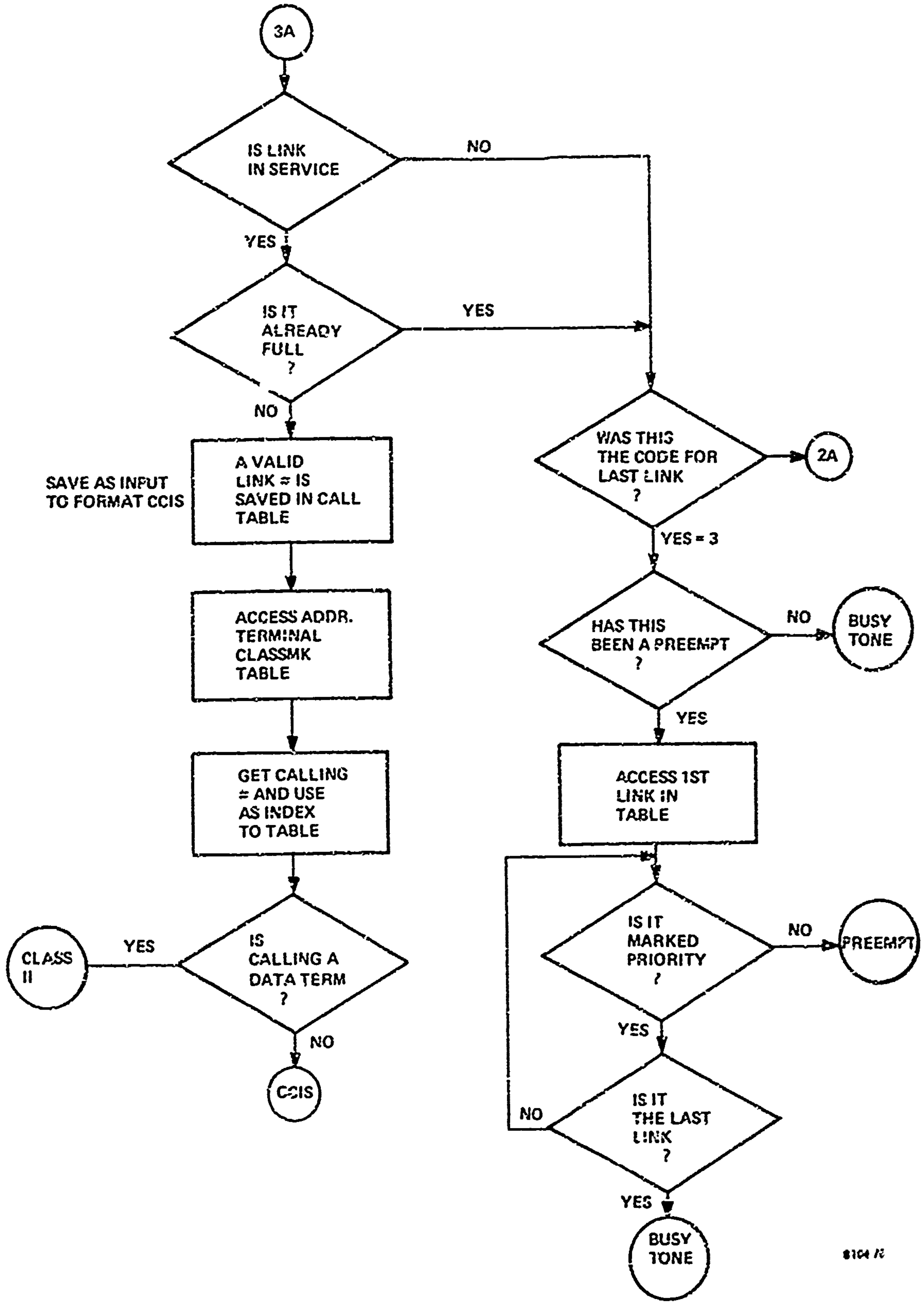


Figure I-4. Translation and Routing (Sheet 3 of 3)

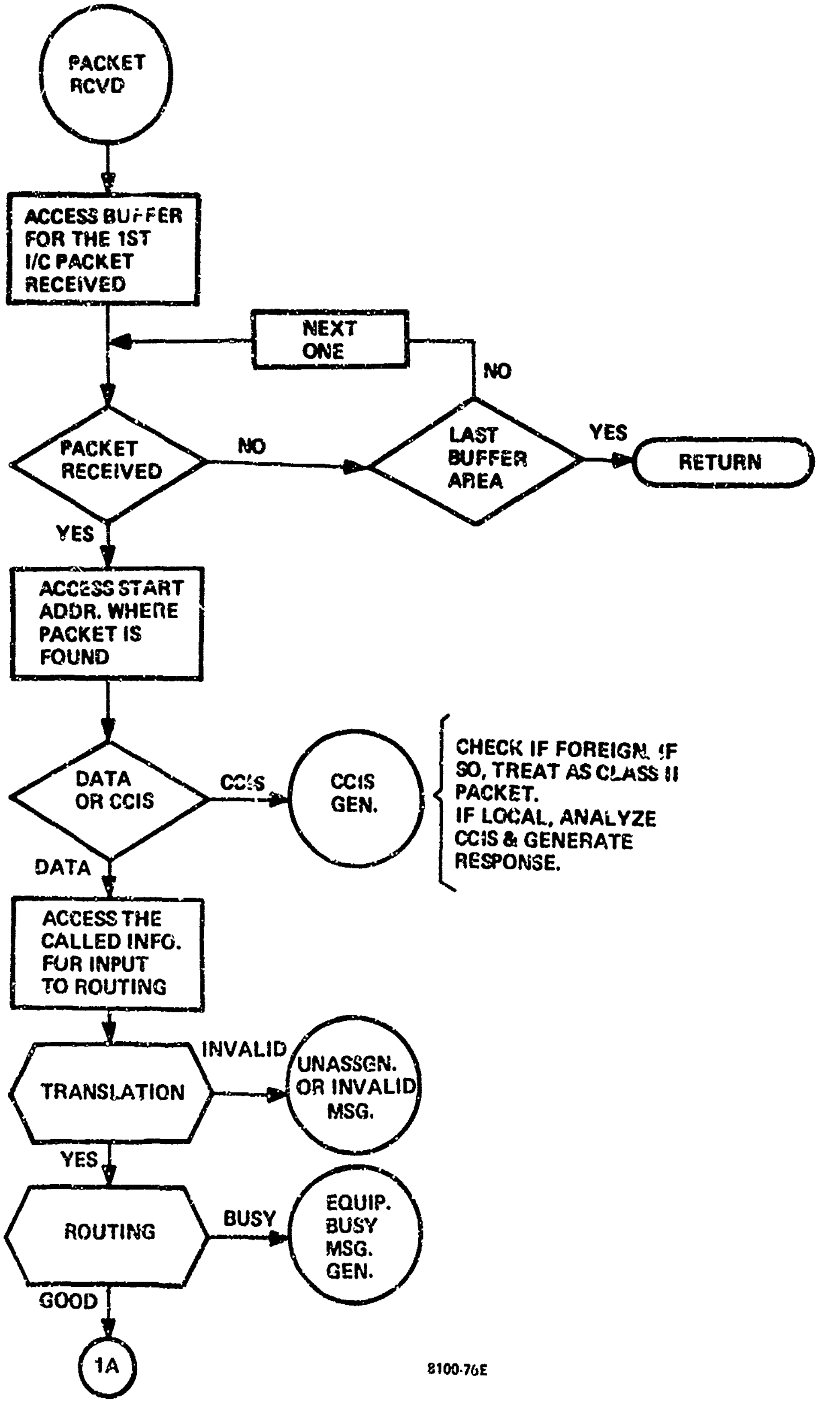

PACKET RCVD
ACCESS BUFFER FOR THE 1ST I/C PACKET RECEIVED
NEXT ONE
NO
PACKET RECEIVED
NO
LAST BUFFER AREA
YES
RETURN
YES
ACCESS START ADDR. WHERE PACKET IS FOUND
DATA OR CCIS
CCIS
CCIS GEN.
CHECK IF FOREIGN. IF SO, TREAT AS CLASS II PACKET.
IF LOCAL, ANALYZE CCIS & GENERATE RESPONSE.
DATA
ACCESS THE CALLED INFO. FOR INPUT TO ROUTING
TRANSLATION
INVALID
UNASSGN. OR INVALID MSG.
YES
ROUTING
BUSY
EQUIP. BUSY MSG. GEN.
GOOD
1A
8100-76E

Figure I-5. LIP Packet Analysis (Sheet 1 of 2)

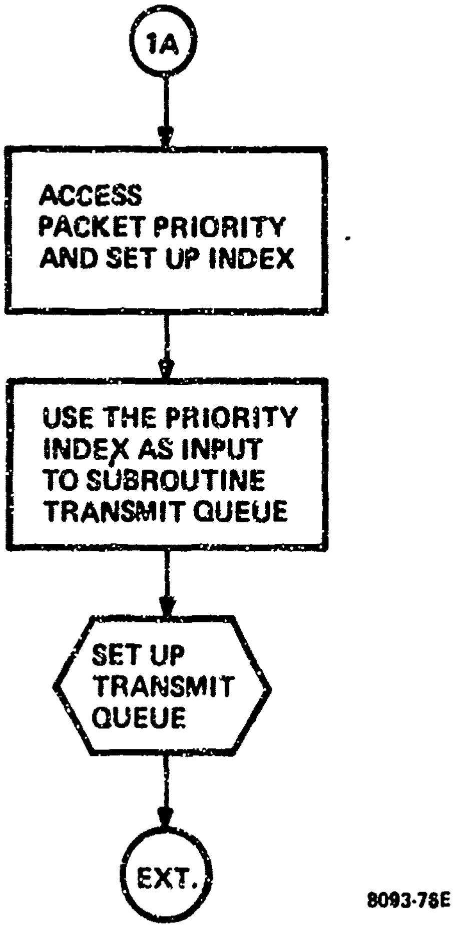

1A
ACCESS PACKET PRIORITY AND SET UP INDEX
USE THE PRIORITY INDEX AS INPUT TO SUBROUTINE TRANSMIT QUEUE
SET UP TRANSMIT QUEUE
EXT.
8093-78E

Figure I-5. LIP Packet Analysis (Sheet 2 of 2)

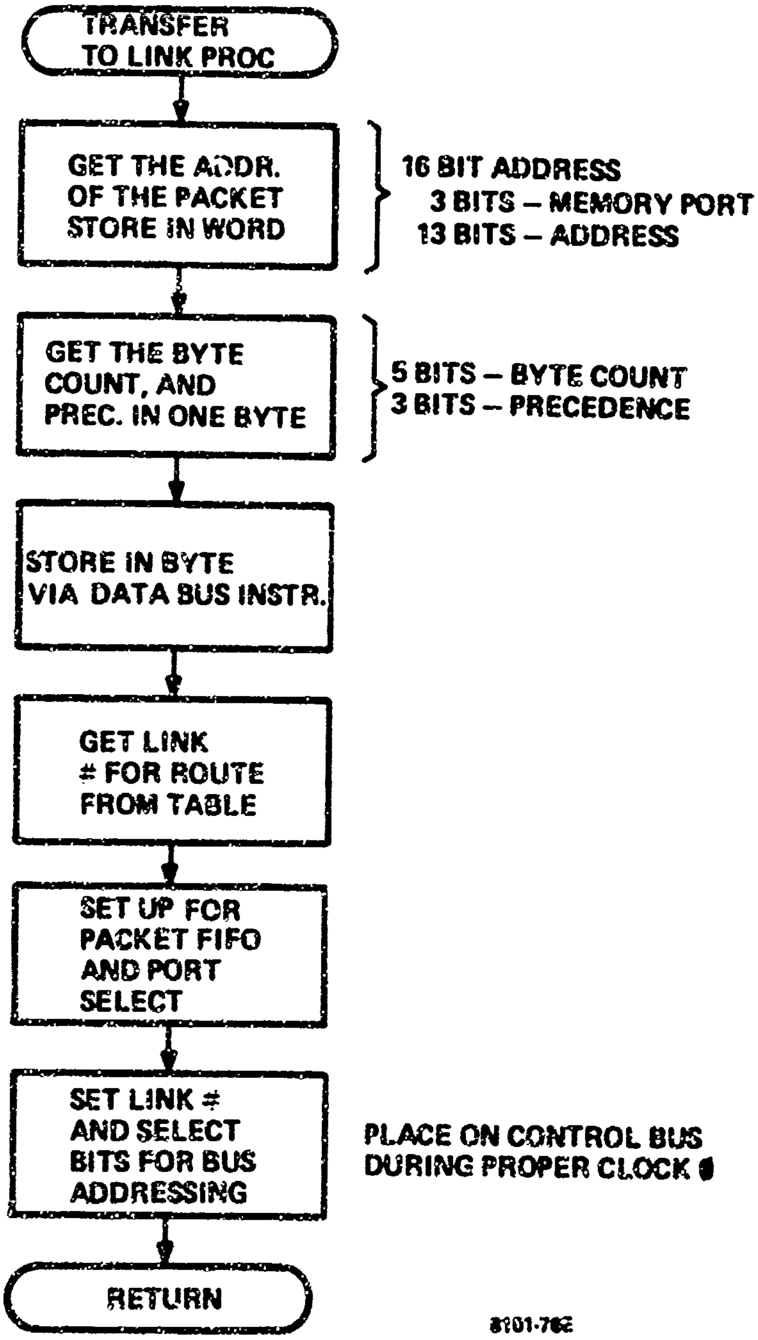

TRANSFER TO LINK PROC
GET THE ADDR. OF THE PACKET STORE IN WORD
16 BIT ADDRESS
3 BITS – MEMORY PORT
13 BITS – ADDRESS
GET THE BYTE COUNT, AND PREC. IN ONE BYTE
5 BITS – BYTE COUNT
3 BITS – PRECEDENCE
STORE IN BYTE VIA DATA BUS INSTR.
GET LINK # FOR ROUTE FROM TABLE
SET UP FOR PACKET FIFO AND PORT SELECT
SET LINK # AND SELECT BITS FOR BUS ADDRESSING
PLACE ON CONTROL BUS DURING PROPER CLOCK Φ
RETURN
8101-762

Figure I-6. Control Switching

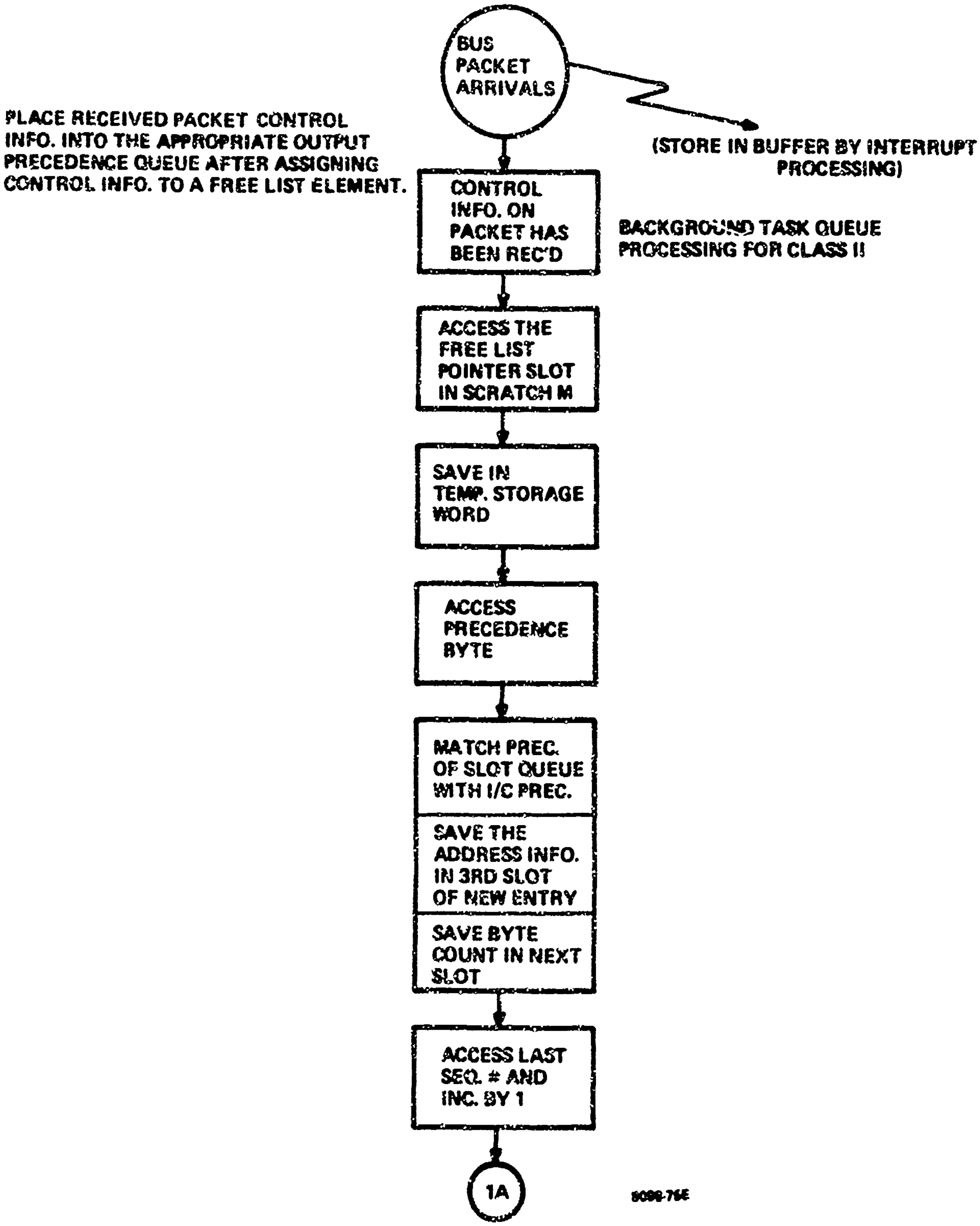

BUS PACKET ARRIVALS
(STORE IN BUFFER BY INTERRUPT PROCESSING)
PLACE RECEIVED PACKET CONTROL INFO. INTO THE APPROPRIATE OUTPUT PRECEDENCE QUEUE AFTER ASSIGNING CONTROL INFO. TO A FREE LIST ELEMENT.
CONTROL INFO. ON PACKET HAS BEEN REC'D
BACKGROUND TASK QUEUE PROCESSING FOR CLASS II
ACCESS THE FREE LIST POINTER SLOT IN SCRATCH M
SAVE IN TEMP. STORAGE WORD
ACCESS PRECEDENCE BYTE
MATCH PREC. OF SLOT QUEUE WITH I/C PREC.
SAVE THE ADDRESS INFO. IN 3RD SLOT OF NEW ENTRY
SAVE BYTE COUNT IN NEXT SLOT
ACCESS LAST SEQ. # AND INC. BY 1
1A
9099-75E

Figure I-7. Output Processor Control (Sheet 1 of 2)

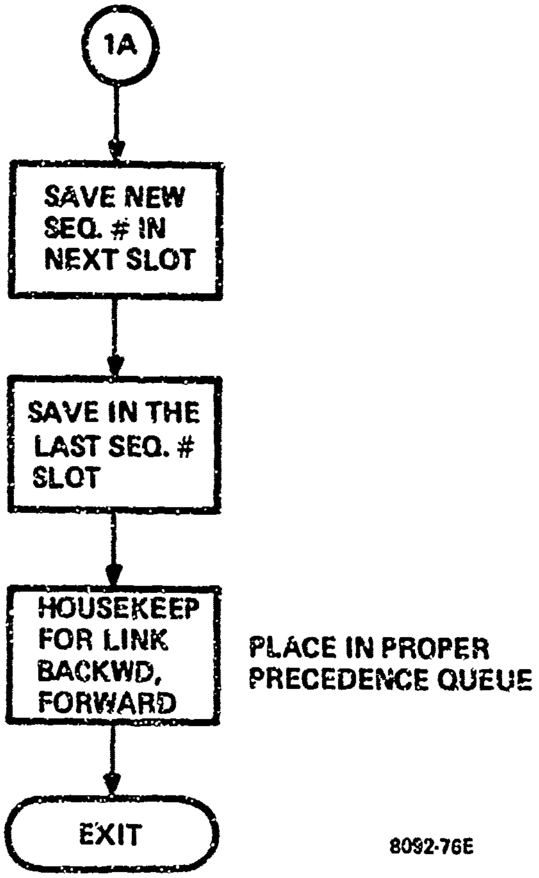

1A
SAVE NEW SEQ. # IN NEXT SLOT
SAVE IN THE LAST SEQ. # SLOT
HOUSEKEEP FOR LINK BACKWD, FORWARD
PLACE IN PROPER PRECEDENCE QUEUE
EXIT
8092-76E

Figure I-7. Output Processor Control (Sheet 2 of 2)

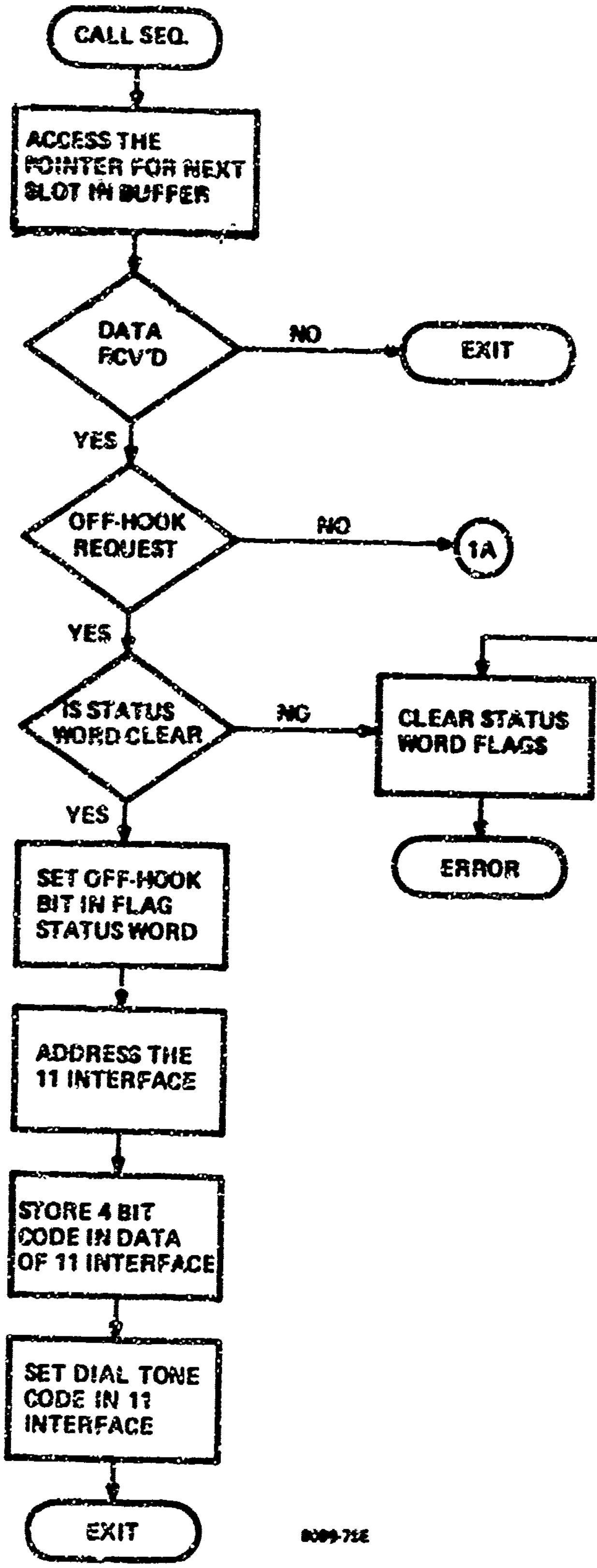
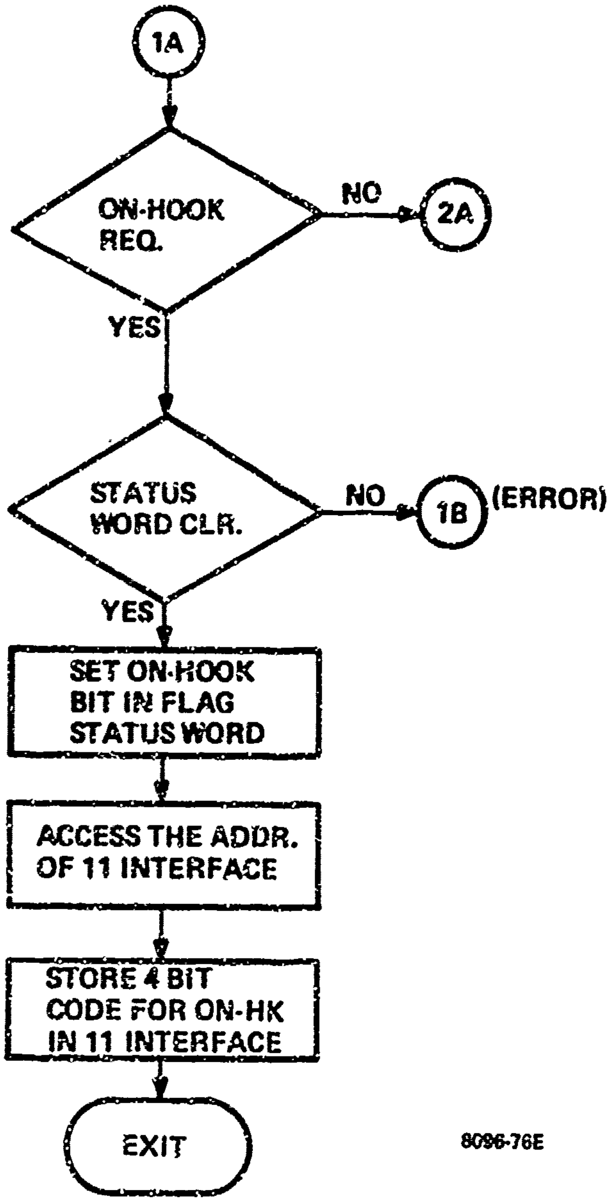

CALL SEQ.
ACCESS THE POINTER FOR NEXT SLOT IN BUFFER
DATA RCV'D
NO
EXIT
YES
OFF-HOOK REQUEST
NO
1A
YES
1B
IS STATUS WORD CLEAR
NO
CLEAR STATUS WORD FLAGS
YES
ERROR
SET OFF-HOOK BIT IN FLAG STATUS WORD
ADDRESS THE 11 INTERFACE
STORE 4 BIT CODE IN DATA OF 11 INTERFACE
SET DIAL TONE CODE IN 11 INTERFACE
EXIT
8009-75E

Figure I-8. Access Processor - Supervision/ Signaling Scan (Sheet 1 of 5)


1A
ON-HOOK REQ.
NO
2A
YES
STATUS WORD CLR.
NO
1B
(ERROR)
YES
SET ON-HOOK BIT IN FLAG STATUS WORD
ACCESS THE ADDR. OF 11 INTERFACE
STORE 4 BIT CODE FOR ON-HK IN 11 INTERFACE
EXIT

8096-76E

Figure I-8. Access Processor - Supervision/ Signaling Scan (Sheet 2 of 5)

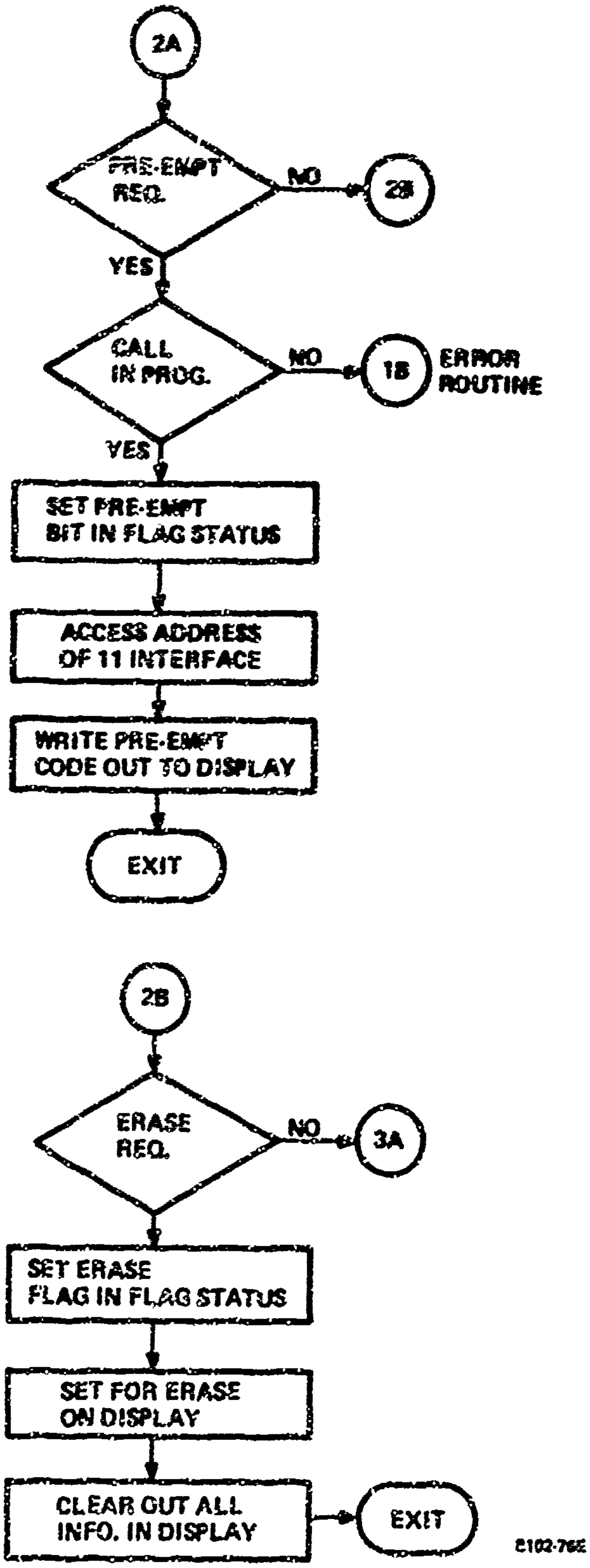

2A
PRE-EMPT REQ.
NO
2B
YES
CALL IN PROG.
NO
1B
ERROR ROUTINE
YES
SET PRE-EMPT BIT IN FLAG STATUS
ACCESS ADDRESS OF 11 INTERFACE
WRITE PRE-EMPT CODE OUT TO DISPLAY
EXIT
2B
ERASE REQ.
NO
3A
SET ERASE FLAG IN FLAG STATUS
SET FOR ERASE ON DISPLAY
CLEAR OUT ALL INFO. IN DISPLAY
EXIT
E102-76E

Figure I-8. Access Processor - Supervision/ Signaling Scan (Sheet 3 of 5)

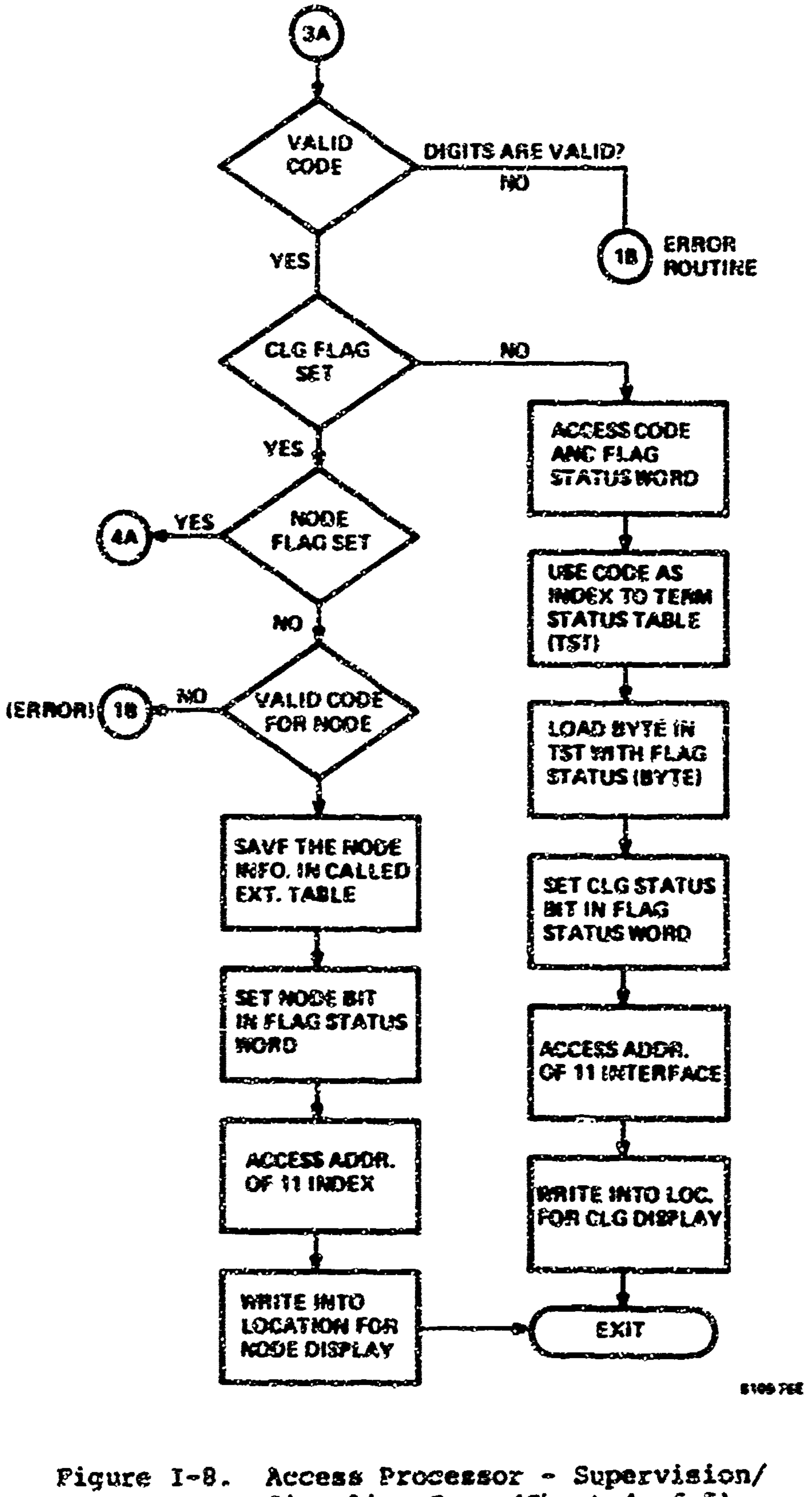

3A
VALID CODE
DIGITS ARE VALID?
NO
1B
ERROR ROUTINE
YES
CLG FLAG SET
NO
ACCESS CODE ANC FLAG STATUS WORD
YES
4A
YES
NODE FLAG SET
USE CODE AS INDEX TO TERM STATUS TABLE (TST)
NO
(ERROR)
1B
NO
VALID CODE FOR NODE
LOAD BYTE IN TST WITH FLAG STATUS (BYTE)
SAVF THE NODE INFO. IN CALLED EXT. TABLE
SET CLG STATUS BIT IN FLAG STATUS WORD
SET NODE BIT IN FLAG STATUS WORD
ACCESS ADDR. OF 11 INTERFACE
ACCESS ADDR. OF 11 INDEX
WRITE INTO LOC. FOR CLG DISPLAY
WRITE INTO LOCATION FOR NODE DISPLAY
EXIT
8109 76E

Figure I-8. Access Processor - Supervision/
Signaling Scan (Sheet 4 of 5)

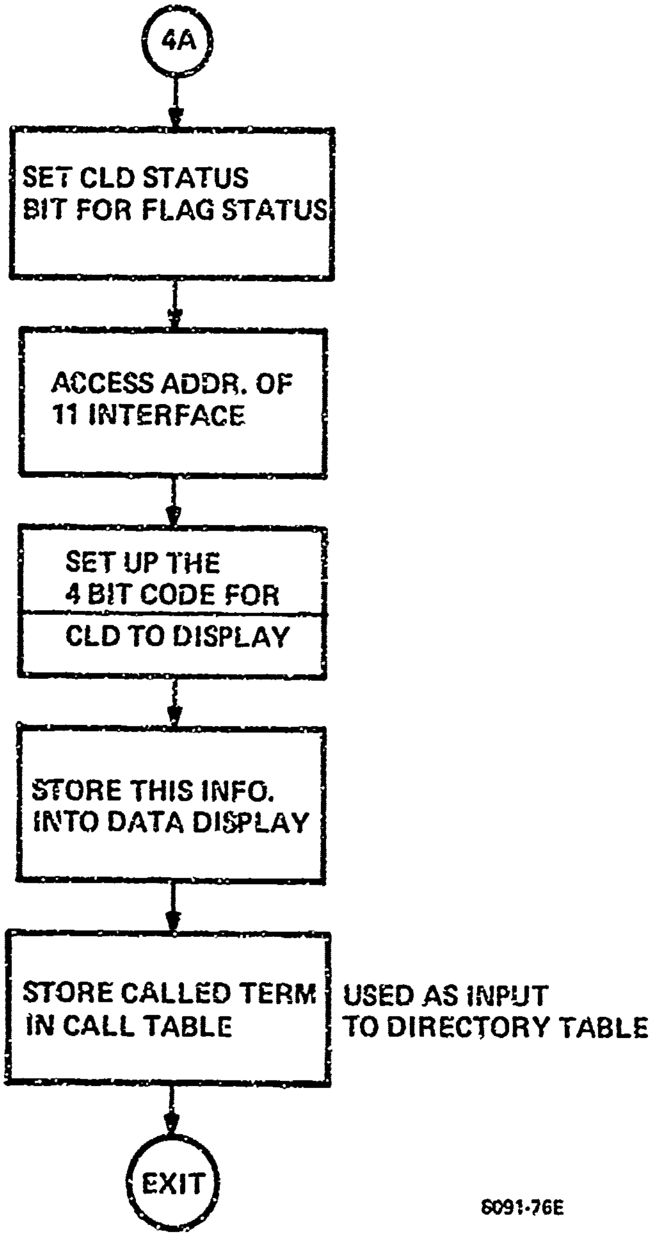


Figure I-8. Access Processor - Supervision/
Signaling Scan (Sheet 5 of 5)

APPENDIX II

DETAILED COST ESTIMATES

TABLE II-1. DETAILED COST ESTIMATES: DIGITAL EQUIPMENT CORPORATION
LSI-11 WITH 3 TTY/NODE

QTY	ITEM	LIST PRICE	PART NO.	SYSTEM USAGE(*)	DISCOUNT (%)	COST(*)
<u>Access Equipment Group</u>						
2	TTY	1300	-	6	0	2600
2	LSI-11 & 4 KW RAM	1696	11/Ø3-EA	10	15	2883
4	4KwRAM	469	MSVII-B	30	32	1276
2	Expander Box	808	BAII-ME	10	32	1099
2	Jumper Cable	234	BCVIB-Ø4	10	32	318
1	SLI	235	DLVII	4	32	160
2	DLVII Cable	40	BCØ5M-IF	8	32	54
13	PLI	129	DRVII	54	32	1140
13	DRVII Cable	32	BCØ4Z-Ø	54	32	283
1	REVII-C	300	REVII-C	4	32	204
2	TEVII	85	TEVII	10	32	116
2	4Kb Data Memory	450	-	-	0	900
12	Boards & Hardware	270	-	-	0	3240
1	Card Nest	350	-	-	0	350
2	4K PROM Card	149	MRVII-AA	10	32	203
64	512X4 PROM Chips	22	MRVII-AC	320	32	957
	Access Equipment Group Total Cost					$15783.
<u>Node Power Supply Cabinets</u>						
1	Power Supply	200	-	-	0	200
1	Cabinet Rack**	550	-	-	0	550
	Node Power Supply Cabinet Total Cost					$750.

* Item cost and system usage is based on quantities needed for a 2-node network.

** 2 needed per node for network greater than 2 nodes.

TABLE II-1. DETAILED COST ESTIMATES: DIGITAL EQUIPMENT CORPORATION LSI-11 WITH 3 TTY/NODE (Cont)

		LIST		SYSTEM	DISCOUNT	
QTY	ITEM	PRICE	PART NO.	USAGE(*)	(%)	COST(*)
	Link Termination Group:					
2	LSI-11 & 4 Kw RAM	1696	11/Ø3-EA	10	15	2883
5	4Kw RAM	469	MSV11-B	30	32	1595
2	Expander Box	808	BA11-ME	10	32	1099
2	Jumper Cable	234	BCV1B-Ø4	10	32	318
11	IOC	129	DRVII	54	32	965
11	DRVII Cable	32	BCØ48-1Ø	54	32	239
1	REVII-C	300	REVII-C	4	32	204
2	TEVII	85	TEVII	10	32	116
2	4Kb Data Memory	450	-	-	0	900
7	Boards & Hardware	270	-	-	0	1890
1	Card Nest	350	-	-	0	350
2	4K PROM Card	149	MRVII-AA	10	32	203
64	512x4 PROM Chips	22	MRVII-AC	320	32	957
	Link Termination Group Total Cost					$11719.
	Nodal Processor Group:					
1	LSI-11 & 4Kw RAM	1696	11/Ø3-EA	10	15	1442
6	4Kw RAM	469	MSVII-B	30	32	1914
3	IOC	129	DRVII	54	32	263
3	DRVII Cable	32	BCØ48-1Ø	54	32	65
1	DLVII (SLI)	235	DLVII	4	32	160
2	DLVII Cable	40	BCØ5M-1F	8	32	54
1	TEVII	85	TEVII	10	32	58
1	Expander Box	808	BA11-ME	10	32	549
1	Jumper Cable	234	BCV1B-Ø4	10	32	159
2	Boards & Hardware	270	-	-	0	540
1	TTY	1300	-	6	0	1300
1	4K PROM Card	149	MRVII-AA	10	32	101
32	512x4 PROM Chips	22	MRVII-AC	320	32	479
	Nodal Processor Group Total Cost					$7084.

*Item cost and system image is based on quantities needed for a 2-node network.

TABLE II-1. DETAILED COST ESTIMATES: DIGITAL EQUIPMENT CORPORATION
LSI-II WITH 3 TTY/NODE (Cont)

QTY	ITEM	LIST PRICE	PART NO.	SYSTEM USAGE(*)	DISCOUNT (%)	COST(*)
DEC Off-Line Software Development System:						
1	System Basics	8883	11VØ3-EA	1	15	7551
3	SLI	235	DLVII	7	32	479
3	DLVII Cable	34	BCØ5M-1F	11	32	69
4	4Kw RAM	469	MSVII-B	34	32	1276
1	DECprinter	3585	LA180-CA	1	32	2438
1	DECwriter	2350	LA36-DE	1	32	1598
1	BASIC/RT-11	500	QJ921-AY	1	32	340

DEC Off-Line Software Development System Total Cost $13751.

*Based on a 2-node network.

TABLE II-2. DETAILED COST ESTIMATES: DATA GENERAL CORPORATION, μNOVA WITH 3 TTY/NODE

QTY	ITEM	LIST PRICE	PART NO.	SYSTEM USAGE(*)	DISCOUNT (%)	COST(*)
	<u>Access Equipment Group:</u>					
2	TTY	1300	-	-	0	2600
1	μNOVA & 4 Kw RAM (9 slot)	1995	8561	10	26	1476
1	μNOVA & 4 Kw RAM (18 slot)	2595	8560	10	26	1920
2	8 Kw RAM	950	8573	26	29	1349
13	IOC Bd.	250	4210	54	36	2080
13	IO Cables (3'-48 cond.)	21	-	-	0	273
2	SLI	250	4207	8	23	385
2	SLI Cables (6'-6 cond.)	5	-	8	0	10
2	4Kb Data Memory	450	-	-	0	900
12	Boards & Hardware	270	-	-	0	3240
1	Card Nest	350	-	-	0	350
2	4Kw PROM	750	-	26	29	1065
	Access Equipment Group Total Cost:					$15,648.
	<u>Node Power Supply & Cabinet:</u>					
1	Power Supply	200	-	-	0	200
1	Cabinet Rack**	550	-	-	0	550
	Node Power Supply and Cabinet Total Cost					$750.
	<u>Link Termination Group:</u>					
2	μNOVA & 4 Kw RAM (9 slot)	1995	8561	10	26	2953
2	8 Kw RAM	950	8573	26	29	1349
1	4 Kw RAM	600	8572	26	29	426
11	IOC Bd.	250	4210	54	36	1760
11	IO Cables	21	-	-	0	231
2	4Kb Data Memory	450	-	-	0	900
7	Boards & Hardware	270	-	-	0	1890
1	Card Nest	350	-	-	0	350
2	4Kw PROM	750	-	26	29	1065
	Link Termination Group Total Cost					$10,924.

* Based on 2-Node Network.

** 2 needed per node when servicing 2 or more links.

TABLE II-2. DETAILED COST ESTIMATES: DATA GENERAL CORPORATION, μNOVA WITH 3 TTY/NODE (Cont)

		LIST		SYSTEM	DISCOUNT	
QTY	ITEM	PRICE	PART NO.	USAGE(*)	(%)	COST(*)
Nodal Processor Group:						
1	μNOVA & 4 Kw RAM (9 slot)	1995	8561	10	26	1476
3	8 Kw RAM	950	8573	26	29	2024
3	IOC Bd.	250	4210	54	36	480
3	IO Cables	21	-	-	0	63
2	SLI	250	4207	3	23	385
2	SLI Cables	5	-	8	0	10
2	Boards & Hardware	270	-	-	0	540
1	TTY	1300	-	-	0	1300
1	4 Kw PROM	750	-	26	29	533
	Nodal Processor Group Total Cost					$6,811.

* Based on a 2-node network.

Data General Off-Line Software

Development System:

A Data General Sales Engineer has indicated that an off-line software development system, having a capability similar to the DEC development system for $13,731.00 would cost approximately $20,000 to $25,000. An average of $22,500 has been used here.